On Matrices With Displacement Structure: Generalized Operators and Faster Algorithms

Alin Bostan, Claude-Pierre Jeannerod, Christophe Mouilleron, Eric Schost

ON MATRICES WITH DISPLACEMENT STRUCTURE: GENERALIZED OPERATORS AND FASTER ALGORITHMS

A. BOSTAN*, C.-P. JEANNEROD†, C. MOUILLERON‡, AND É. SCHOST§

Abstract. For matrices with displacement structure, basic operations like multiplication, inversion, and linear system solving can all be expressed in terms of the following task: evaluate the product AB, where A is a structured $n \times n$ matrix of displacement rank α, and B is an arbitrary $n \times \alpha$ matrix. Given B and a so-called *generator* of A, this product is classically computed with a cost ranging from $O(\alpha^2\mathscr{M}(n))$ to $O(\alpha^2\mathscr{M}(n)\log(n))$ arithmetic operations, depending on the type of structure of A; here, $\mathscr{M}$ is a cost function for polynomial multiplication. In this paper, we first generalize classical displacement operators, based on block diagonal matrices with companion diagonal blocks, and then design fast algorithms to perform the task above for this extended class of structured matrices. The cost of these algorithms ranges from $O(\alpha^{\omega-1}\mathscr{M}(n))$ to $O(\alpha^{\omega-1}\mathscr{M}(n)\log(n))$, with ω such that two $n \times n$ matrices over a field can be multiplied using $O(n^\omega)$ field operations. By combining this result with classical randomized regularization techniques, we obtain faster Las Vegas algorithms for structured inversion and linear system solving.

Key words. structured linear algebra, matrix multiplication, computational complexity

AMS subject classifications. 65F05, 68Q25

1. Introduction.

Exploiting the structure of data is key to develop fast algorithms and, in the context of linear algebra, this principle is at the heart of algorithms for *displacement structured matrices*. These algorithms can speed up for instance the inversion of a given matrix whenever this matrix has a structure close to that of a Toeplitz, Hankel, Vandermonde, or Cauchy matrix. The idea is to represent structured matrices succinctly by means of their *generators* with respect to suitable *displacement operators*, and to operate on this succinct data structure.

Displacement operators. Let $\mathbb{F}$ be a field. To measure the extent to which a matrix $\mathsf{A} \in \mathbb{F}^{m\times n}$ possesses some structure, it is customary to use its *displacement rank*, that is, the rank of its image through a *displacement operator* [18]. There exist two broad classes of displacement operators: *Sylvester operators*, of the form

$$\nabla_{\mathsf{M},\mathsf{N}} : \mathsf{A} \in \mathbb{F}^{m\times n} \mapsto \mathsf{MA} - \mathsf{AN} \in \mathbb{F}^{m\times n},$$

and *Stein operators*, of the form

$$\Delta_{\mathsf{M},\mathsf{N}} : \mathsf{A} \in \mathbb{F}^{m\times n} \mapsto \mathsf{A} - \mathsf{MAN} \in \mathbb{F}^{m\times n};$$

in both cases, M and N are fixed matrices in $\mathbb{F}^{m\times m}$ and $\mathbb{F}^{n\times n}$, respectively. For any such operator, say $\mathcal{L}$, the rank of $\mathcal{L}(\mathsf{A})$ is called the $\mathcal{L}$-*displacement rank* of A, or simply its displacement rank if $\mathcal{L}$ is clear from the context. Loosely speaking, the matrix A is called *structured* (with respect to the operator $\mathcal{L}$) if its $\mathcal{L}$-displacement rank is small compared to its sizes m and n.

We say that a matrix pair (G,H) in $\mathbb{F}^{m\times\alpha} \times \mathbb{F}^{n\times\alpha}$ is an $\mathcal{L}$-*generator of length* α of $\mathsf{A} \in \mathbb{F}^{m\times n}$ if it satisfies $\mathcal{L}(A) = \mathsf{GH}^t$, with H^t the transpose of H. Again, if $\mathcal{L}$ is clear

*Inria, France (alin.bostan@inria.fr).

†Inria, Université de Lyon, laboratoire LIP (CNRS, ENSL, Inria, UCBL), France (claude-pierre.jeannerod@inria.fr).

‡ENSIIE, Évry, France (christophe.mouilleron@ens-lyon.org). Part of this work was done when C. Mouilleron was a member of laboratoire LIP (CNRS, ENSL, Inria, UCBL).

§David R. Cheriton School of Computer Science, University of Waterloo, ON, Canada (eschost@uwaterloo.ca).

from the context, we shall simply say *generator of length* α. Provided $\mathcal{L}$ is invertible and when α is small, such a generator can play the role of a succinct data structure to represent the matrix A. The smallest possible value for α is the rank of $\mathcal{L}(\mathsf{A})$.

If A is invertible and structured with respect to $\mathcal{L}$, then its inverse A^{-1} is structured with respect to the operator $\mathcal{L}'$ obtained by swapping M and N:

$$\mathcal{L} = \nabla_{\mathsf{M},\mathsf{N}} \implies \mathcal{L}' = \nabla_{\mathsf{N},\mathsf{M}}, \qquad \mathcal{L} = \Delta_{\mathsf{M},\mathsf{N}} \implies \mathcal{L}' = \Delta_{\mathsf{N},\mathsf{M}}. \tag{1}$$

More precisely, the above claim says that the ranks of $\mathcal{L}(\mathsf{A})$ and $\mathcal{L}'(\mathsf{A}^{-1})$ are the same. (See [33, Theorem 1.5.3] and, for the case not handled there, see [18, p. 771]; for completeness, we give a proof in Appendix A.)

Some classical operators. Consider first Toeplitz-like matrices. For φ in $\mathbb{F}$, it is customary to define the $m \times m$ cyclic down-shift matrix

$$\mathbb{Z}_{m,\varphi} = \begin{bmatrix} & & & \varphi \\ 1 & & & \\ & \ddots & & \\ & & 1 & \end{bmatrix} \in \mathbb{F}^{m\times m}. \tag{2}$$

Then, using Sylvester operators, a matrix $\mathsf{A} \in \mathbb{F}^{m\times n}$ will be called Toeplitz-like if $\mathbb{Z}_{m,\varphi}\,\mathsf{A} - \mathsf{A}\,\mathbb{Z}_{n,\psi}$ has a low rank compared to m and n. (This rank is independent of the choice of φ and ψ, up to an additive constant of absolute value at most two.) Using Stein operators, one would consider instead the rank of $\mathsf{A} - \mathbb{Z}_{m,\varphi}\,\mathsf{A}\,\mathbb{Z}_{n,\psi}^t$.

Besides $\mathbb{Z}_{m,\varphi}$ and its transpose it is also useful to consider the diagonal matrix $\mathbb{D}(\mathsf{x})$, whose diagonal is given by $\mathsf{x} \in \mathbb{F}^m$. Specifically, popular choices take

$$(\mathsf{M},\mathsf{N}) \in \left\{\mathbb{Z}_{m,\varphi}, \mathbb{Z}_{m,\varphi}^t, \mathbb{D}(\mathsf{x})\right\} \times \left\{\mathbb{Z}_{n,\psi}, \mathbb{Z}_{n,\psi}^t, \mathbb{D}(\mathsf{y})\right\}, \tag{3}$$

with $\varphi, \psi \in \mathbb{F}$, $\mathsf{x} \in \mathbb{F}^m$ and $\mathsf{y} \in \mathbb{F}^n$, for either Sylvester or Stein operators. This family covers in particular Toeplitz and Hankel-like structures, where both M and N are (transposed) cyclic down-shift matrices; Vandermonde-like structures, where one of the matrices M and N is a (transposed) cyclic down-shift matrix and the other one is diagonal; and Cauchy-like structures, where both M and N are diagonal. We say that a displacement operator is of *Toeplitz / Hankel type* if it falls into the first category, and of *Vandermonde / Cauchy type* if it falls into the second or third one.

Block-diagonal companion matrices. The first contribution of this paper is to generalize the classical displacement operators seen before. Let $\mathbb{F}[x]$ be the univariate polynomial ring over $\mathbb{F}$, and for $\delta \geqslant 0$, let $\mathbb{F}[x]_\delta$ be the $\mathbb{F}$-vector space of polynomials of degree less than δ. Furthermore, for $F = \sum_i f_i x^i \in \mathbb{F}[x]$ monic of degree $\delta > 0$, denote by $\mathbb{M}_F$ the $\delta \times \delta$ companion matrix associated with F:

$$\mathbb{M}_F = \begin{bmatrix} & & & -f_0 \\ 1 & & & -f_1 \\ & \ddots & & \vdots \\ & & 1 & -f_{\delta-1} \end{bmatrix} \in \mathbb{F}^{\delta\times\delta}.$$

Two special cases are the $m \times m$ matrix $\mathbb{Z}_{m,\varphi}$ defined in (2) and the 1×1 matrix given by $x_0 \in \mathbb{F}$, which are associated with the polynomials $x^m - \varphi$ and $x - x_0$, respectively.

Given $d > 0$, let now $\mathbf{P} = P_1, \ldots, P_d$ denote a family of monic nonconstant polynomials in $\mathbb{F}[x]$, and let $m = m_1 + \cdots + m_d$ with $m_i = \deg(P_i) \geqslant 1$ for all i. We

will call *block-diagonal companion matrix* associated with $\mathbf{P}$ the $m \times m$ block-diagonal matrix $\mathbb{M}_{\mathbf{P}}$ whose ith diagonal block is the $m_i \times m_i$ companion matrix $\mathbb{M}_{P_i}$:

$$\mathbb{M}_{\mathbf{P}} = \begin{bmatrix} \mathbb{M}_{P_1} & & \\ & \ddots & \\ & & \mathbb{M}_{P_d} \end{bmatrix} \in \mathbb{F}^{m \times m}.$$

We write P to denote the product $P_1 \cdots P_d$, which is the characteristic polynomial of $\mathbb{M}_{\mathbf{P}}$. Finally, we associate with $\mathbf{P}$ the following assumption on its elements:

$$\mathscr{H}_{\mathbf{P}}: \quad P_1, \ldots, P_d \text{ are pairwise coprime.}$$

Associated displacement operators. With $\mathbf{P}$ as above, we now consider another family of monic nonconstant polynomials, namely $\mathbf{Q} = Q_1, \ldots, Q_e$, with respective degrees $n_1, \ldots, n_e$; we let $Q = Q_1 \cdots Q_e$ and $n = n_1 + \cdots + n_e$. In what follows, we assume that both assumptions $\mathscr{H}_{\mathbf{P}}$ and $\mathscr{H}_{\mathbf{Q}}$ hold.

Let then $\mathbb{M}_{\mathbf{P}} \in \mathbb{F}^{m \times m}$ and $\mathbb{M}_{\mathbf{Q}} \in \mathbb{F}^{n \times n}$ be the block-diagonal companion matrices associated with $\mathbf{P}$ and $\mathbf{Q}$. In this paper, we consider the following eight operators:

$$\nabla_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}_{\mathbf{Q}}^t}, \quad \nabla_{\mathbb{M}_{\mathbf{P}}^t, \mathbb{M}_{\mathbf{Q}}}, \quad \nabla_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}_{\mathbf{Q}}}, \quad \nabla_{\mathbb{M}_{\mathbf{P}}^t, \mathbb{M}_{\mathbf{Q}}^t},$$

$$\Delta_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}_{\mathbf{Q}}^t}, \quad \Delta_{\mathbb{M}_{\mathbf{P}}^t, \mathbb{M}_{\mathbf{Q}}}, \quad \Delta_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}_{\mathbf{Q}}}, \quad \Delta_{\mathbb{M}_{\mathbf{P}}^t, \mathbb{M}_{\mathbf{Q}}^t}.$$

We shall call them the *operators associated with* $(\mathbf{P}, \mathbf{Q})$ *of Sylvester or Stein type*, respectively, and say that they have *format* (m, n). Two of these operators will be highlighted, $\nabla_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}_{\mathbf{Q}}^t}$ and $\Delta_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}_{\mathbf{Q}}^t}$, and we will call them the *basic* operators associated with $(\mathbf{P}, \mathbf{Q})$. Indeed, we will be able to derive convenient formulas to invert them; this will in turn allow us to deal with the other six operators by suitable reductions.

Tables 1 and 2 show that the classical structures defined by (3) follow as special cases of these operators, for suitable choices of $(\mathbf{P}, \mathbf{Q})$ making $\mathbb{M}_{\mathbf{P}}$ or $\mathbb{M}_{\mathbf{Q}}$ a diagonal or a (transposed) cyclic down-shift matrix.

TABLE 1
Some particular cases for the Sylvester operator $\nabla_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}_{\mathbf{Q}}^t}$.

	$e = 1$ and $Q_1 = x^n - \psi$	$e = n$ and $Q_j = x - y_j$
$d = 1$ and $P_1 = x^m - \varphi$	Hankel-like	Vandermonde^t-like (for the points $1/y_j$)
$d = m$ and $P_i = x - x_i$	Vandermonde-like (for the points $1/x_i$)	Cauchy-like (for the points x_i and y_j)

TABLE 2
Some particular cases for the Stein operator $\Delta_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}_{\mathbf{Q}}^t}$.

	$e = 1$ and $Q_1 = x^n - \psi$	$e = n$ and $Q_j = x - y_j$
$d = 1$ and $P_1 = x^m - \varphi$	Toeplitz-like	Vandermonde^t-like (for the points y_j)
$d = m$ and $P_i = x - x_i$	Vandermonde-like (for the points x_i)	Cauchy-like (for the points $1/x_i$ and y_j)

In other words, our family of operators covers all classical operators of Toeplitz, Hankel, Vandermonde, and Cauchy type in a unified manner. However, this formalism also includes many new cases as "intermediate" situations. For example, taking $e = 1$ and $Q_1 = x^n$, we see that the matrix of the *multiple reduction map* modulo $P_1, \ldots, P_d$ has displacement rank 1 for the Stein operator $\Delta_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}^t_{\mathbf{Q}}}$, so our operators will allow us to address problems related to Chinese remaindering.

Problem statement. Our goal is to study the complexity of basic computations with matrices that are structured for the operators seen above: multiply a matrix $\mathsf{A} \in \mathbb{F}^{m\times n}$ by a matrix B, invert A, and solve the linear system $\mathsf{Ax} = \mathsf{b}$. Formally, let $\mathcal{L} : \mathbb{F}^{m\times n} \to \mathbb{F}^{m\times n}$ be a displacement operator of either Sylvester or Stein type, and assume that $\mathcal{L}$ is invertible. Then, the problems we address read as follows.

$\mathsf{mul}(\mathcal{L}, \alpha, \beta)$: given an $\mathcal{L}$-generator $(\mathsf{G}, \mathsf{H}) \in \mathbb{F}^{m\times\alpha} \times \mathbb{F}^{n\times\alpha}$ of a matrix $\mathsf{A} \in \mathbb{F}^{m\times n}$ with $\alpha \leqslant \min(m, n)$ and given a matrix $\mathsf{B} \in \mathbb{F}^{n\times\beta}$, compute the product $\mathsf{AB} \in \mathbb{F}^{m\times\beta}$.

$\mathsf{inv}(\mathcal{L}, \alpha)$: given an $\mathcal{L}$-generator $(\mathsf{G}, \mathsf{H}) \in \mathbb{F}^{m\times\alpha} \times \mathbb{F}^{m\times\alpha}$ of a matrix $\mathsf{A} \in \mathbb{F}^{m\times m}$ with $\alpha \leqslant m$, return an $\mathcal{L}'$-generator $(\mathsf{G}', \mathsf{H}')$ of length at most α for A^{-1}, or assert that A is not invertible. Here, $\mathcal{L}'$ is the operator in (1).

$\mathsf{solve}(\mathcal{L}, \alpha)$: given an $\mathcal{L}$-generator $(\mathsf{G}, \mathsf{H}) \in \mathbb{F}^{m\times\alpha} \times \mathbb{F}^{n\times\alpha}$ of a matrix $\mathsf{A} \in \mathbb{F}^{m\times n}$ with $\alpha \leqslant \min(m, n)$ and given a vector $\mathsf{b} \in \mathbb{F}^m$, find a solution $\mathsf{x} \in \mathbb{F}^n$ to $\mathsf{Ax} = \mathsf{b}$, or assert that no such solution exists. (When $\mathsf{b} = 0$ and A has not full column rank, then a nonzero vector x should be returned; we call this a *nontrivial solution*.)

For the problems above to make sense, we need the displacement operator $\mathcal{L}$ to be invertible. For the Sylvester operators associated with $(\mathbf{P}, \mathbf{Q})$, this occurs if and only if $\gcd(P, Q) = 1$; for Stein operators, the condition becomes $\gcd(P, \widetilde{Q}) = 1$ with $\widetilde{Q} = x^n Q(1/x)$ or, equivalently, $\gcd(\widetilde{P}, Q) = 1$ with $\widetilde{P} = x^m P(1/x)$ [33, Theorem 4.3.2].

We allow probabilistic algorithms. For instance, for inversion in size m, we will see algorithms that use $r = O(m)$ random elements in $\mathbb{F}$, for which success is conditional to avoiding a hypersurface in $\overline{\mathbb{F}}^r$ of degree $m^{O(1)}$.

To analyze all upcoming algorithms, we count all arithmetic operations in $\mathbb{F}$ at unit cost, so the *time* spent by an algorithm is simply the number of such operations it performs. Then, our goal is to give upper bounds on the cost functions $\mathscr{C}_{\mathrm{mul}}(.,.,.)$, $\mathscr{C}_{\mathrm{solve}}(.,.)$, $\mathscr{C}_{\mathrm{inv}}(.,.)$, which are such that the problems $\mathsf{mul}(\mathcal{L}, \alpha, \beta)$, $\mathsf{solve}(\mathcal{L}, \alpha)$, $\mathsf{inv}(\mathcal{L}, \alpha)$, can be solved in respective times

$$\mathscr{C}_{\mathrm{mul}}(\mathcal{L}, \alpha, \beta), \quad \mathscr{C}_{\mathrm{solve}}(\mathcal{L}, \alpha), \quad \mathscr{C}_{\mathrm{inv}}(\mathcal{L}, \alpha).$$

In particular, $\mathscr{C}_{\mathrm{mul}}(\mathcal{L}, \alpha, 1)$ denotes the cost of a structured matrix-vector product Av.

A few further problems can be solved as direct extensions of these operations. Indeed, a simple transformation (see e.g. [33, Theorem 1.5.2]) shows that if a matrix A is structured with respect to one of the operators associated with $(\mathbf{P}, \mathbf{Q})$, its transpose A^t is structured as well, with respect to one of the operators associated with $(\mathbf{Q}, \mathbf{P})$. This is summarized as follows, where $(\mathsf{M}, \mathsf{N}) \in \{\mathbb{M}_{\mathbf{P}}, \mathbb{M}^t_{\mathbf{P}}\} \times \{\mathbb{M}_{\mathbf{Q}}, \mathbb{M}^t_{\mathbf{Q}}\}$:

$$\nabla_{\mathsf{M},\mathsf{N}}(\mathsf{A}) = \mathsf{GH}^t \iff \nabla_{\mathsf{N}^t,\mathsf{M}^t}(\mathsf{A}^t) = (-\mathsf{H})\,\mathsf{G}^t,$$
$$\Delta_{\mathsf{M},\mathsf{N}}(\mathsf{A}) = \mathsf{GH}^t \iff \Delta_{\mathsf{N}^t,\mathsf{M}^t}(\mathsf{A}^t) = \mathsf{H}\,\mathsf{G}^t.$$

Thus, from our results on multiplication, inversion, and linear system solving for the matrix A, one directly obtains similar results for the same operations with A^t.

Cost measures. We let $\mathscr{M}$ be a multiplication time function for $\mathbb{F}[x]$, that is, $\mathscr{M} : \mathbb{N}_{>0} \to \mathbb{R}_{>0}$ is such that two polynomials of degree less than d can be multiplied in $\mathscr{M}(d)$ arithmetic operations in $\mathbb{F}$; $\mathscr{M}$ must also satisfy the super-linearity

properties of [12, Ch. 8]. It follows from [38, 37] that we can always take $\mathscr{M}(d) \in O(d\lg(d)\lglg(d))$, where $\lg(d) = \log(\max(d,2))$ and log is the binary logarithm. (This definition of $\lg(d)$ ensures that expressions like $d\lg(d)\lglg(d)$ do not vanish at $d=1$.) Hence, from now on we assume that $\mathscr{M}$ is quasi-linear in d, so that $\mathscr{M}(d) = O(d^{1+\epsilon})$ for all $\epsilon > 0$, and that $\mathscr{M}(1) = 1$.

Here and hereafter, ω denotes any real number such that two $n \times n$ matrices over $\mathbb{F}$ can be multiplied using $O(n^\omega)$ arithmetic operations in $\mathbb{F}$. We have $\omega < 2.38$ [23] as $n \to \infty$ and, for moderate dimensions (say, $n < 10^6$), upper bounds on ω range from 2.81 [40] down to 2.78 [29]; on the other hand, we have the trivial lower bound $\omega \geqslant 2$. (Our cost analyses will be given for $\omega > 2$ for simplicity, but note that their extension to the case $\omega = 2$ would be straightforward and imply no more than some extra logarithmic factors.)

The running time of our algorithms will depend on the cost of some polynomial matrix operations. Let $\mathscr{M}_{\mathrm{mat}}(d,n) : \mathbb{R}_{\geqslant 1} \times \mathbb{N}_{>0} \to \mathbb{R}_{>0}$ be such that two $n \times n$ matrices over $\mathbb{F}[x]_d$ can be multiplied in $\mathscr{M}_{\mathrm{mat}}(d,n)$ arithmetic operations in $\mathbb{F}$. (It will be convenient to allow non-integer values of d; this does not change the definition.) One can use the following estimates:

$$\mathscr{M}_{\mathrm{mat}}(d,n) = \begin{cases} O(\mathscr{M}(d)n^\omega) & \text{in general,} \\ O(dn^\omega + \mathscr{M}(d)n^2) & \text{if } \mathrm{char}(\mathbb{F}) = 0 \text{ or } |\mathbb{F}| \geqslant 2d; \end{cases}$$

the former follows from [9] and the latter is from [7], using evaluation and interpolation at geometric sequences; in both cases, this is $\widetilde{O}(dn^\omega)$, where the soft-O notation $\widetilde{O}(\cdot)$ means that we omit polylogarithmic factors. Our costs will be expressed using not quite $\mathscr{M}_{\mathrm{mat}}$, but two related functions $\mathscr{M}'_{\mathrm{mat}}$ and $\mathscr{M}''_{\mathrm{mat}}$, which should be thought of as being "close" to $\mathscr{M}_{\mathrm{mat}}$ up to logarithmic factors. These two functions are defined as follows. For n a power of two,

$$\mathscr{M}'_{\mathrm{mat}}(d,n) = \sum_{k=0}^{\log(n)} 2^k \mathscr{M}_{\mathrm{mat}}\left(2^k d, \frac{n}{2^k}\right)$$

and, for general n, $\mathscr{M}'_{\mathrm{mat}}(d,n) = \mathscr{M}'_{\mathrm{mat}}(d,\overline{n})$ with $\overline{n}$ the smallest power of two greater than or equal to n. Using the super-linearity of $\mathscr{M}$, the above choices for $\mathscr{M}_{\mathrm{mat}}$ give

$$\mathscr{M}'_{\mathrm{mat}}(d,n) = \begin{cases} O(\mathscr{M}(dn)n^{\omega-1}) & \text{in general,} \\ O(dn^\omega + \mathscr{M}(dn)n\lg(n)) & \text{if } \mathrm{char}(\mathbb{F}) = 0 \text{ or } |\mathbb{F}| \geqslant 2dn; \end{cases} \tag{4}$$

in both cases, this is again $\widetilde{O}(dn^\omega)$. Note on the other hand that $dn^2 \leqslant n\,\mathscr{M}(dn) \leqslant \mathscr{M}'_{\mathrm{mat}}(d,n)$. Now, for d a power of two, let

$$\mathscr{M}''_{\mathrm{mat}}(d,n) = \sum_{k=0}^{\log(d)} 2^k \mathscr{M}'_{\mathrm{mat}}\left(\frac{d}{2^k}, n\right)$$

and, for general d, let $\mathscr{M}''_{\mathrm{mat}}(d,n) = \mathscr{M}''_{\mathrm{mat}}(\bar{d},n)$ with $\bar{d}$ the smallest power of two greater than or equal to d. Using the two previous bounds on $\mathscr{M}'_{\mathrm{mat}}$, we deduce that

$$\mathscr{M}''_{\mathrm{mat}}(d,n) = \begin{cases} O\big(\mathscr{M}(dn)n^{\omega-1}\lg(d)\big) & \text{in general,} \\ O\big((dn^\omega + \mathscr{M}(dn)n\lg(n))\lg(d)\big) & \text{if } \mathrm{char}(\mathbb{F}) = 0 \text{ or } |\mathbb{F}| \geqslant 2dn; \end{cases}$$

again, in both cases this is $\widetilde{O}(dn^\omega)$. Conversely, we have $dn^\omega = O\big(\mathscr{M}''_{\mathrm{mat}}(d,n)\big)$, since $\mathscr{M}''_{\mathrm{mat}}(d,n) \geqslant d\,\mathscr{M}'_{\mathrm{mat}}(1,n) \geqslant d\,\mathscr{M}_{\mathrm{mat}}(1,n)$. Remark also that if we assume, similarly to the super-linearity assumptions on $\mathscr{M}$, that for every positive integer n the function $d \mapsto \mathscr{M}'_{\mathrm{mat}}(d,n)/d$ is nondecreasing, then we have in all cases the simple bound

$$\mathscr{M}''_{\mathrm{mat}}(d,n) \leqslant \mathscr{M}'_{\mathrm{mat}}(d,n)(1+\log(d)).$$

Finally, note that for $d = 1$ and $\omega > 2$,

$$\mathscr{M}''_{\mathrm{mat}}(1,n) = \mathscr{M}'_{\mathrm{mat}}(1,n) = O(n^\omega).$$

To check the second equality, one may fix some ϵ such that $0 < \epsilon < \omega - 2$, so that $\mathscr{M}(m) = O(m^{1+\epsilon})$ and thus $\mathscr{M}'_{\mathrm{mat}}(1,n) = O\big(\sum_{k\geqslant 0} n^\omega (2^k)^{2+\epsilon-\omega}\big)$ with $2+\epsilon-\omega < 0$.

To any family of polynomials $\mathbf{P} = P_1, \ldots, P_d$ with $m_i = \deg(P_i) > 0$ for all i, we will associate two cost measures, $\mathscr{C}(\mathbf{P})$ and $\mathscr{D}(\mathbf{P})$, related to Chinese remaindering and (extended) GCDs; both range between $O(m)$ or $O(\mathscr{M}(m))$ and $O(\mathscr{M}(m)\mathrm{lg}(m))$, where we write $m = m_1 + \cdots + m_d$ as before. In both cases, we give a general formula and point out particular cases with smaller estimates.

First, in the particular case where $P_i = x - x_i$ for all i with the x_i in geometric progression, we take $\mathscr{C}(\mathbf{P}) = \mathscr{M}(m)$. In all other cases, we take

$$\mathscr{C}(\mathbf{P}) = \mathscr{M}(m)\left(1 + \sum_{1\leqslant i\leqslant d} \frac{m_i}{m} \log\left(\frac{m}{m_i}\right)\right).$$

Since the value of the above sum ranges from zero to $\log(d)$ (see for example [8, p. 38]), the function $\mathscr{C}(\mathbf{P})$ always satisfies

$$\mathscr{M}(m) \leqslant \mathscr{C}(\mathbf{P}) \leqslant \mathscr{M}(m)(1+\log(d));$$

in particular, we have $\mathscr{C}(\mathbf{P}) = O(\mathscr{M}(m)\mathrm{lg}(d))$.

The definition of the function $\mathscr{D}(\mathbf{P})$ starts with a particular case as well: when $d = 1$ and $P_1 = x^m - \varphi$ with $\varphi \in \mathbb{F}$, we take $\mathscr{D}(\mathbf{P}) = m$. Otherwise, we write

$$\mathscr{D}(\mathbf{P}) = \sum_{1\leqslant i\leqslant d} \mathscr{M}(m_i)\mathrm{lg}(m_i).$$

Table 3 displays the values of $\mathscr{C}(\mathbf{P})$ and $\mathscr{D}(\mathbf{P})$ in the two special cases mentioned above, but also their estimates when $d = O(1)$ or $m_i = O(1)$ for $1 \leqslant i \leqslant d$. We see that when $d = O(1)$ we are essentially in the best case for $\mathscr{C}(\mathbf{P})$ and in the worst case for $\mathscr{D}(\mathbf{P})$, and that the situation is reversed when $m_i = O(1)$ for all i. In other words, the families with best and worst cases for $\mathscr{D}(\mathbf{P})$ are the opposite of those for $\mathscr{C}(\mathbf{P})$.

Finally, when considering two families of polynomials $\mathbf{P}$ and $\mathbf{Q}$, we will write

$$\mathscr{C}(\mathbf{P},\mathbf{Q}) = \mathscr{C}(\mathbf{P}) + \mathscr{C}(\mathbf{Q}) \qquad \text{and} \qquad \mathscr{D}(\mathbf{P},\mathbf{Q}) = \mathscr{D}(\mathbf{P}) + \mathscr{D}(\mathbf{Q}).$$

Background work. Following landmark publications such as [24, 18, 25, 3], the literature on structured linear systems has vastly developed, and we refer especially to [33, 34] for comprehensive overviews. It was recognized early that a key ingredient for the development of fast algorithms was the ability to perform matrix-vector products efficiently. For Toeplitz-like and Hankel-like systems, this was done in [25, 3];

TABLE 3
Upper bounds for $\mathscr{C}(\mathbf{P})$ and $\mathscr{D}(\mathbf{P})$ in some special cases.

	$\mathscr{C}(\mathbf{P})$	$\mathscr{D}(\mathbf{P})$
$P_i = x - x_i$ for all i, with the x_i in geometric progression	$\mathscr{M}(m)$	m
$d = 1$ and $P_1 = x^m - \varphi$	$\mathscr{M}(m)$	m
$m_i = O(1)$ for all i	$O(\mathscr{M}(m)\lg(m))$	$O(m)$
$d = O(1)$	$O(\mathscr{M}(m))$	$O(\mathscr{M}(m)\lg(m))$

for Vandermonde and Cauchy structures, this is in [13, 14]. In our notation, these references establish bounds of the form

$$\mathscr{C}_{\mathrm{mul}}(\mathcal{L}, \alpha, 1) = O(\alpha\mathscr{M}(m)) \quad \text{or} \quad \mathscr{C}_{\mathrm{mul}}(\mathcal{L}, \alpha, 1) = O(\alpha\mathscr{M}(m)\lg(m)),$$

for operators of format (m, m) of Toeplitz / Hankel type or Vandermonde / Cauchy type, respectively.

If a matrix B has β columns, then the product AB can be computed by performing β matrix-vector products with A. In other words, we can take

$$\mathscr{C}_{\mathrm{mul}}(\mathcal{L}, \alpha, \beta) \leqslant \beta\,\mathscr{C}_{\mathrm{mul}}(\mathcal{L}, \alpha, 1),$$

which yields the bounds $O(\alpha\beta\mathscr{M}(m))$ and $O(\alpha\beta\mathscr{M}(m)\lg(m))$ for the cases seen above.

Such matrix-matrix products, with $\beta \simeq \alpha$, are the main ingredients in the Morf / Bitmead-Anderson (MBA) algorithm to invert structured matrices [3, 25] of Toeplitz / Hankel type, which combines the displacement rank approach of [11, 18, 17] with Strassen's divide-and-conquer approach for dense matrix inversion [40]. This algorithm initially required several genericity conditions to hold but, based on the randomized regularization technique of Kaltofen and Saunders [21], it was then further extended in [19, 20] to handle arbitrary matrices (see also [2, pp. 204–208] and [33, §5.5–5.7]). For operators of format (m, m) of Toeplitz, Hankel, Vandermonde or Cauchy type, the cost of inversion $\mathscr{C}_{\mathrm{inv}}(\mathcal{L}, \alpha)$ can then (roughly speaking) be taken in

$$O(\mathscr{C}_{\mathrm{mul}}(\mathcal{L}, \alpha, \alpha)\lg(m));$$

see [36] for the case of Vandermonde-like and Cauchy-like matrices, and [26, 31, 32] for a unified treatment of all four structures. Using the above upper bound on the cost of multiplication $\mathscr{C}_{\mathrm{mul}}(\mathcal{L}, \alpha, \alpha)$, we see that this is $\alpha\lg(m)$ times the cost of a matrix-vector product (strictly speaking, some precautions are necessary to make such a statement; for instance, the algorithm becomes probabilistic).

This results in running times of the form $O(\alpha^2\mathscr{M}(m)\lg(m))$ or $O(\alpha^2\mathscr{M}(m)\lg(m)^2)$ for inverting respectively Toeplitz / Hankel-like matrices and Vandermonde / Cauchy-like matrices of size m and displacement rank α. In the Vandermonde / Cauchy case, another approach due to Pan [30] proceeds by reduction to the Toeplitz / Hankel case; the running time is then reduced to $O(\alpha^2\mathscr{M}(m)\lg(m))$.

Going beyond the four basic structures, Olshevsky and Shokrollahi introduced a common generalization thereof, where the displacement matrices take the form of block-diagonal Jordan matrices [28]. We will discuss their result in more detail after

stating our main theorems. For the moment, we mention that their paper gives an algorithm for matrix-vector product with running times ranging from

$$\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,1)=O(\alpha\mathscr{M}(m)) \quad \text{to} \quad \mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,1)=O(\alpha\mathscr{M}(m)\mathrm{lg}(m)),$$

depending on the block configuration of the supporting Jordan matrices; for the four basic structures seen above, we recover the running times seen before. In [27], this result is used to sketch an extension of the MBA algorithm to such matrices. As before, the running time for inversion increases by a factor $\alpha\mathrm{lg}(m)$ compared to the time for matrix-vector multiplication, ranging from

$$\mathscr{C}_{\mathrm{inv}}(\mathcal{L},\alpha)=O(\alpha^2\mathscr{M}(m)\mathrm{lg}(m)) \quad \text{to} \quad \mathscr{C}_{\mathrm{inv}}(\mathcal{L},\alpha)=O(\alpha^2\mathscr{M}(m)\mathrm{lg}(m)^2),$$

depending on the operator.

When $\alpha=O(1)$, all above results for matrix-vector product or matrix inversion are within a polylogarithmic factor of being linear-time. However, when α is not constant, this is not the case anymore; in the worst case where $\alpha\simeq m$, the running times for inversion grow like m^3 (neglecting logarithmic factors again), whereas dense linear algebra algorithms take time $O(m^\omega)$. In [4], Bostan, Jeannerod and Schost gave algorithms for inversion of structured matrices of the four classical types with running time $O(\alpha^{\omega-1}\mathscr{M}(m)\mathrm{lg}(m)^2)$. This is satisfactory for $\alpha\simeq m$, since we recover the cost of dense linear algebra, up to logarithmic factors. However, for $\alpha=O(1)$, this algorithm is slightly slower (by a factor $\mathrm{lg}(m)$) than the MBA algorithm.

Main results. The results in this paper cover in a uniform manner the general class of operators based on companion matrices introduced above. Inspired by [4], we obtain algorithms for structured matrix multiplication, inversion, and linear system solving whose running times grow with α as $\alpha^{\omega-1}$ instead of α^2; however, we manage to avoid the loss of the $\mathrm{lg}(m)$ factor observed in [4], thus improving on the results of that paper. All our algorithms assume exact arithmetic (and we do not claim any kind of accuracy guarantee when using finite precision arithmetic). Note also that they are deterministic for multiplication and, similarly to [4], randomized (Las Vegas) for inversion and linear system solving.

Let $\mathbf{P}$ and $\mathbf{Q}$ be as before. We give in Section 3 inversion formulas for the basic operators $\mathcal{L}=\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}$ or $\mathcal{L}=\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}$ that allow us to recover a matrix A from one of its generators, generalizing simultaneously previous results for Toeplitz, Hankel, Vandermonde, and Cauchy displacement operators. From these formulas, one readily deduces that a matrix-vector product $\mathsf{A}\mathsf{u}$ can be computed in time

$$\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,1)=O(\mathscr{D}(\mathbf{P},\mathbf{Q})+\alpha\mathscr{C}(\mathbf{P},\mathbf{Q})),$$

which is $\widetilde{O}(\alpha p)$ for $p=\max(m,n)$.

For matrix-matrix products, the direct approach which is based on the estimate $\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,\beta)\leqslant\beta\,\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,1)$ thus leads to $\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,\beta)=\widetilde{O}(\alpha\beta p)$, which is sub-optimal, as we pointed out before. The following theorem improves on this straightforward approach, for all operators associated with $(\mathbf{P},\mathbf{Q})$.

THEOREM 1. *Let $\mathbf{P}=P_1,\ldots,P_d$ and $\mathbf{Q}=Q_1,\ldots,Q_e$ be two families of polynomials in $\mathbb{F}[x]$. Assume that in each of these families the polynomials are monic, non-constant and pairwise coprime, and denote $m=\sum_{i=1}^d\deg(P_i)$ and $n=\sum_{i=1}^e\deg(Q_i)$. Then, for any invertible operator $\mathcal{L}$ associated with $(\mathbf{P},\mathbf{Q})$, we can take*

$$\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,\beta)=O\left(\frac{\beta'}{\alpha'}\mathscr{M}'_{\mathrm{mat}}\left(\frac{p}{\alpha'},\alpha'\right)+\mathscr{D}(\mathbf{P},\mathbf{Q})+\beta'\mathscr{C}(\mathbf{P},\mathbf{Q})\right),$$

with $p = \max(m,n)$, $\alpha' = \min(\alpha,\beta)$, *and* $\beta' = \max(\alpha,\beta)$. *Furthermore,*

$$\mathscr{C}_{\mathrm{inv}}(\mathcal{L},\alpha),\ \mathscr{C}_{\mathrm{solve}}(\mathcal{L},\alpha) \ = \ O\left(\mathscr{M}''_{\mathrm{mat}}\left(\frac{p}{\alpha},\alpha\right)\right).$$

Using the estimates for $\mathscr{M}'_{\mathrm{mat}}$ given in (4), the estimate for $\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,\beta)$ becomes

$$\begin{cases} O\left({\alpha'}^{\omega-2}\beta'\mathscr{M}(p) + \beta'\mathscr{M}(p)\lg(p)\right) & \text{in general,} \\ O\left({\alpha'}^{\omega-2}\beta' p + \beta'\mathscr{M}(p)\lg(p)\right) & \text{if } \mathrm{char}(\mathbb{F}) = 0 \text{ or } |\mathbb{F}| \geqslant 2p. \end{cases}$$

Thus, these results provide a continuum between two extreme cases. When $\alpha = \beta = O(1)$, the cost is $O(\mathscr{M}(p)\lg(p))$; when α and β are large, the first term is dominant, with total costs respectively $O({\alpha'}^{\omega-2}\beta'\mathscr{M}(p))$ and $O({\alpha'}^{\omega-2}\beta' p)$. Disregarding logarithmic factors, this is always $\widetilde{O}({\alpha'}^{\omega-2}\beta' p)$: this matches the cost (up to logarithmic factors) of dense, unstructured linear algebra algorithms for multiplying matrices of dimensions $\alpha' \times p$ and $p \times \beta'$ with $\alpha' \leqslant \min(p,\beta')$.

In fact, when $\alpha = p \leqslant \beta$ we recover exactly the cost bound $O(\beta p^{\omega-1})$ of the multiplication of two dense unstructured matrices of dimensions $p \times p$ and $p \times \beta$, since then $\mathscr{C}_{\mathrm{mul}}(\mathcal{L},p,\beta) = O\big(\frac{\beta}{p}\mathscr{M}'_{\mathrm{mat}}(1,p) + \mathsf{M}(p)\lg(p)\big)$ and $\mathscr{M}'_{\mathrm{mat}}(1,p) = O(p^\omega)$.

For inversion and system solving, the bounds given above on $\mathscr{M}''_{\mathrm{mat}}$ yield

$$\begin{cases} O\big(\alpha^{\omega-1}\mathscr{M}(p)\lg(p)\big) & \text{in general,} \\ O\big(\alpha^{\omega-1}p\lg(p) + \alpha\lg(\alpha)\mathscr{M}(p)\lg(p)\big) & \text{if } \mathrm{char}(\mathbb{F}) = 0 \text{ or } |\mathbb{F}| \geqslant 2p; \end{cases}$$

and, when $\alpha = p$ (so that the input matrix is $p \times p$ and not structured with respect to the operator $\mathcal{L}$), the obtained cost is $O(\mathscr{M}''_{\mathrm{mat}}(1,p)) \subset O(p^\omega)$, as can be expected.

The following theorem highlights situations where we can take both $\mathscr{C}(\mathbf{P},\mathbf{Q})$ and $\mathscr{D}(\mathbf{P},\mathbf{Q})$ in $O(\mathscr{M}(p))$. In this case, we get slightly better bounds than in the general case for the problems mul (for inversion and solve, the terms above are always negligible, for any choice of $\mathbf{P}$ and $\mathbf{Q}$). In view of Tables 1 and 2, we see that these special cases correspond to operators of Toeplitz / Hankel type, or Vandermonde / Cauchy type, when their supports are points in geometric progression.

THEOREM 2. *All notation and assumptions being as in Theorem 1, suppose in addition that one of the following holds:*

- *either* $d = 1$ *and* $P_1 = x^m - \varphi$ *for some* $\varphi \in \mathbb{F}$, *or* $d = m$ *and there exist* $u, q \in \mathbb{F}$ *such that* $P_i = x - uq^i$ *for all* i*;*
- *either* $e = 1$ *and* $Q_1 = x^n - \psi$ *for some* $\psi \in \mathbb{F}$, *or* $e = n$ *and there exist* $v, r \in \mathbb{F}$ *such that* $Q_j = x - vr^j$ *for all* j.

Then, for any invertible operator $\mathcal{L}$ *associated with* $(\mathbf{P},\mathbf{Q})$, *we can take*

$$\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,\beta) = O\left(\frac{\beta'}{\alpha'}\mathscr{M}'_{\mathrm{mat}}\left(\frac{p}{\alpha'},\alpha'\right) + \beta'\mathscr{M}(p)\right),$$

with $p = \max(m,n)$, $\alpha' = \min(\alpha,\beta)$, *and* $\beta' = \max(\alpha,\beta)$.

Using once again the bounds on $\mathscr{M}'_{\mathrm{mat}}$ given in (4), this is thus

$$\begin{cases} O({\alpha'}^{\omega-2}\beta'\mathscr{M}(p)) & \text{in general,} \\ O({\alpha'}^{\omega-2}\beta' p + \beta'\lg(\alpha')\mathscr{M}(p)) & \text{if } \mathrm{char}(\mathbb{F}) = 0 \text{ or } |\mathbb{F}| \geqslant 2p. \end{cases}$$

Comparison with previous work. Let us briefly compare our results with previous ones, in increasing order of generality.

For classical operators of Hankel, Toeplitz, Vandermonde or Cauchy type, our results match classical ones in the cases $\alpha = \beta = O(1)$ (for multiplication) or $\alpha = O(1)$ (for inversion). When we drop such assumptions, our results improve on all previous ones. For instance, for the inversion of Toeplitz-like or Hankel-like matrices, the best previous results were either $O(\alpha^2 \mathscr{M}(m)\lg(m))$ in [25, 3, 19], or $O(\alpha^{\omega-1}\mathscr{M}(m)\lg(m)^2)$ in [4]. We improve them simultaneously, by dropping the cost to $O(\alpha^{\omega-1}\mathscr{M}(m)\lg(m))$.

We mentioned earlier the work of Olshevsky and Shokrollahi [27, 28], who consider Sylvester operators where the displacement matrices are block-diagonal, with Jordan blocks. A Jordan block of size m associated with $\lambda \in \mathbb{F}$ is similar to the companion matrix of $(x-\lambda)^m$; the similarity matrix is the "Pascal" matrix of the mapping $F(x) \mapsto F(x+\lambda)$, for $F \in \mathbb{F}[x]_m$. Because the similarity matrix and its inverse can be applied in time $O(\mathscr{M}(m))$ to a vector [1], the problem considered by Olshevsky and Shokrollahi is essentially equivalent to a particular case of ours.

As it turns out, for these particular cases, the results in [27, 28] for matrix / vector product and inversion are equivalent to ours, when the displacement rank α is $O(1)$. For larger α, our results improve on those of [27, 28], whose costs are quadratic in α.

Finally, to the best of our knowledge, no previous work addressed the general case of block-companion matrices that we deal with in this paper.

An application. We are able to address problems related to *simultaneous approximation* using our operators. Suppose that we are given $\mathbf{P} = P_1, \ldots, P_d$, with sum of degrees m, together with "residuals"

$$R_{1,1}, \ldots, R_{1,\alpha}, \ldots, R_{d,1}, \ldots, R_{d,\alpha},$$

with all $R_{i,j}$ in $\mathbb{F}[x]$ such that $\deg(R_{i,j}) < \deg(P_i)$ for all i, j. We are looking for polynomials $f_1, \ldots, f_\alpha$ in $\mathbb{F}[x]$ such that $f_1 R_{i,1} + \cdots + f_\alpha R_{i,\alpha} = 0 \bmod P_i$ holds for $i = 1, \ldots, d$, and with prescribed degree bounds (the polynomial f_j should be in $\mathbb{F}[x]_{n_j}$, with $n_1 + \cdots + n_\alpha = O(m)$).

The matrix of the corresponding linear system, where the coefficients of $f_1, \ldots, f_\alpha$ are unknown, has size $O(m)$ and displacement rank $O(\alpha)$ for the operator $\Delta_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}^t_{\mathbf{Q}}}$, where $\mathbf{Q}$ is the "vector" consisting of the unique polynomial $Q_1 = x^m - \varphi$, where φ is chosen such that the assumptions of Theorem 1 hold. Hence, we can solve such a system using $O(\alpha^{\omega-1}\mathscr{M}(m)\log(m))$ base field operations. This is to be compared with an algorithm given in [10], that has cost $O((\alpha+d)^{\omega-1}\mathscr{M}(m)\log(m)^2)$, but which does not require the assumption that $P_1, \ldots, P_d$ be pairwise coprime.

This problem generalizes Hermite-Padé (with $d = 1$ and $P_1 = x^m$) and so-called M-Padé approximation, with $P_i = (x - x_i)^{m_i}$, for pairwise distinct x_i in $\mathbb{F}$. A typical instance is the reconstruction of the minimal polynomial of an algebraic function. Suppose that f is a root of a polynomial $S \in \mathbb{F}[x][y]$, so f lies in the algebraic closure of $\mathbb{F}(x)$. Given the power series expansion of f at high enough precision around a point x_1, it is possible to reconstruct S by means of Hermite-Padé approximation. Using our algorithm, we obtain the same output for the same cost, starting this time from approximations at points $x_1, \ldots, x_d$, each of them requiring smaller precision.

Organization of the paper. Section 2 introduces notation used all along this paper, and discusses a few classical questions about polynomial arithmetic, such as Chinese remaindering. Section 3 gives *inversion formulas* for the operators we discuss, which generalize well-known results for classical operators. In Section 4, we use these formulas in order to reduce most of our questions to similar questions for *basic operators*, or

even Toeplitz / Hankel operators. Section 5 gives the main algorithmic result in this paper, an improved algorithm for the multiplication of structured matrices. Finally, in Section 6, we show how this new multiplication algorithm can be used within the MBA approach to accelerate structured inversion and structured system solving.

Acknowledgements. We thank an anonymous referee for many detailed comments.

2. Preliminaries. In this section, we review some classical operations on polynomials and matrices: we introduce a short list of useful matrices, such as multiplication matrices, reversal matrices and Krylov matrices, and we discuss questions related to modular computations, or multiple reduction and Chinese remaindering.

Throughout the article, we will use some well-known complexity results on polynomial arithmetic; as a general rule, they can be found in the books [12], [8], or [33].

2.1. Basic notation. Matrices (resp. vectors) are written in upper-case (resp. lower-case) sans-serif font. If A (resp. B, C, ...) is a matrix, a_i (resp. b_i, c_i, ...) is its ith column. If x (resp. y, z, ...) is a vector, its ith entry is written x_i (resp. $y_i, z_i, \ldots$). Special matrices (cyclic, companion, diagonal, ...) will be written with blackboard bold letters ($\mathbb{Z}$, $\mathbb{M}$, $\mathbb{D}$, ...). The entries of an $m \times n$ matrix A are indexed from 0 to $m-1$ (row indices) and from 0 to $n-1$ (column indices).

We will also use the following notation.

- For $\mathsf{u} = [u_0 \ \cdots \ u_{m-1}]^t \in \mathbb{F}^m$, we write $\mathsf{pol}(\mathsf{u})$ to denote the polynomial $u_0 + \cdots + u_{m-1}x^{m-1} \in \mathbb{F}[x]_m$.
- For any polynomial $F \in \mathbb{F}[x]$ of degree at most d, $\mathsf{rev}(F, d)$ denotes its reverse polynomial $x^d F(1/x) \in \mathbb{F}[x]$.

For $m \geqslant 1$, $\mathbb{J}_m$ is the $m \times m$ reversal matrix, with 1s on the antidiagonal only. More generally, for $1 \leqslant \ell \leqslant m$, $\mathbb{J}_{\ell,m}$ is the $m \times m$ matrix with 1s on the upper antidiagonal entries of indices $(\ell-1, 0), \ldots, (0, \ell-1)$, so $\mathbb{J}_{m,m} = \mathbb{J}_m$; by convention, for $\ell = 0$, $\mathbb{J}_{0,m}$ is the zero matrix of size m.

Finally, for $\mathsf{A} \in \mathbb{F}^{m\times m}$, $\mathsf{v} \in \mathbb{F}^m$, and $\ell \geqslant 1$, we write $\mathbb{K}(\mathsf{A}, \mathsf{v}, \ell)$ to denote the Krylov matrix in $\mathbb{F}^{m\times \ell}$ whose columns are $\mathsf{v}, \mathsf{Av}, \ldots, \mathsf{A}^{\ell-1}\mathsf{v}$.

2.2. Chinese remaindering and related problems. In this subsection, we introduce additional notation and give basic results related to multiple reduction and Chinese remaindering. Consider pairwise-coprime monic polynomials $\mathbf{P} = P_1, \ldots, P_d$ in $\mathbb{F}[x]$ with $\deg(P_i) = m_i$, and let $m = m_1 + \cdots + m_d$ and $P = P_1 \cdots P_d$.

Recall that we associated with the family $\mathbf{P}$ the cost functions $\mathscr{C}(\mathbf{P})$ and $\mathscr{D}(\mathbf{P})$. They will help us measure the cost of the following operations. To $\mathbf{P}$, we associate the *multiple reduction* mapping, and its inverse, *Chinese remaindering*, defined by

$$\begin{array}{rccc} \mathsf{red}_{\mathbf{P}}: & \mathbb{F}[x]_m & \to & \mathbb{F}[x]_{m_1} \times \cdots \times \mathbb{F}[x]_{m_d} \\ & A & \mapsto & (A \bmod P_1, \ldots, A \bmod P_d) \end{array}$$

and

$$\begin{array}{rccc} \mathsf{crt}_{\mathbf{P}}: & \mathbb{F}[x]_{m_1} \times \cdots \times \mathbb{F}[x]_{m_d} & \to & \mathbb{F}[x]_m \\ & (A_1, \ldots, A_d) & \mapsto & A, \text{ such that } A \bmod P_i = A_i \text{ for all } i. \end{array}$$

A related question is *linear recombination*, defined as the following isomorphism:

$$\begin{array}{rccc} \mathsf{comb}_{\mathbf{P}}: & \mathbb{F}[x]_{m_1} \times \cdots \times \mathbb{F}[x]_{m_d} & \to & \mathbb{F}[x]_m \\ & (A_1, \ldots, A_d) & \mapsto & A_1P_2\cdots P_d + \cdots + P_1\cdots P_{d-1}A_d. \end{array}$$

Fast algorithms for these three operations lead to the costs given in the next lemma. Although such costs can be found in or deduced from [8, 12], they do not seem to have

been presented in this way, with a common precomputation involving both functions $\mathscr{C}(\mathbf{P})$ and $\mathscr{D}(\mathbf{P})$, and then an extra cost involving only $\mathscr{C}(\mathbf{P})$.

LEMMA 3. *Let $\mathbf{P} = P_1, \ldots, P_d$ as above be given and assume that $\mathscr{H}_{\mathbf{P}}$ holds. Then, after a precomputation of time $O(\mathscr{C}(\mathbf{P}) + \mathscr{D}(\mathbf{P}))$ that yields P as a by-product, one can apply the maps $\mathsf{red}_{\mathbf{P}}$, its inverse $\mathsf{crt}_{\mathbf{P}}$, as well as $\mathsf{comb}_{\mathbf{P}}$ and its inverse, to any vector in time $O(\mathscr{C}(\mathbf{P}))$.*

Proof. For $1 \leqslant i \leqslant d$, let us define $E_i = P/P_i \bmod P_i$ and $F_i = 1/E_i \bmod P_i$. (In particular, if $d = 1$ then $E_1 = F_1 = 1$, while if P_i is the linear polynomial $x - x_i$ then $E_i = 1/F_i$ is the value of the derivative of P at x_i.) Using these polynomials, we first note how the mappings $\mathsf{crt}_{\mathbf{P}}$ and $\mathsf{comb}_{\mathbf{P}}$ relate to each other: Chinese remaindering of $A_1, \ldots, A_d$ is done by computing $A_iF_i \bmod P_i$ for all i, and then applying $\mathsf{comb}_{\mathbf{P}}$ to these products; conversely, to perform linear recombination it suffices to compute the products $A_iE_i \bmod P_i$ and then to apply $\mathsf{crt}_{\mathbf{P}}$. Note also that the inverse of $\mathsf{comb}_{\mathbf{P}}$ can be obtained by first using $\mathsf{red}_{\mathbf{P}}$ and then multiplying by the F_i's modulo P_i.

Suppose we are in the case where $P_i = x - x_i$ with the x_i's in *geometric* progression. Then, we can compute $P, E_1, \ldots, E_d$ and apply $\mathsf{red}_{\mathbf{P}}$ and $\mathsf{crt}_{\mathbf{P}}$ in time $O(\mathscr{C}(\mathbf{P}))$ using the algorithms of [7]. Using the remark above on the relation between $\mathsf{crt}_{\mathbf{P}}$ and $\mathsf{comb}_{\mathbf{P}}$, the claim carries over to $\mathsf{comb}_{\mathbf{P}}$ and its inverse, so the lemma is proved in this case.

In the general case, we can precompute P, as well as apply $\mathsf{red}_{\mathbf{P}}$ and $\mathsf{comb}_{\mathbf{P}}$ (without any further precomputation) in time $O(\mathscr{C}(\mathbf{P}))$ using respectively Theorems 2.19, 3.19 and Step 2 (or 5) of Theorem 3.21 in [8].

Let $P^\star = P/P_1 + \cdots + P/P_d$. Since $P^\star$ is obtained by applying $\mathsf{comb}_{\mathbf{P}}$ to the polynomials $(1, \ldots, 1)$, it can thus be precomputed in time $O(\mathscr{C}(\mathbf{P}))$. The modular images $P^\star \bmod P_i$, which coincide with the polynomials E_i seen above, can be precomputed in the same time by applying $\mathsf{red}_{\mathbf{P}}$. Finally, we precompute the inverses $F_i = 1/E_i \bmod P_i$ in time $O(\mathscr{D}(\mathbf{P}))$ by fast extended GCD (when $d = 1$, this is straightforward, since we have seen that $E_1 = 1$; otherwise, each F_i is obtained in time $O(\mathscr{M}(m_i)\lg(m_i))$ using the fast extended Euclidean algorithm [8, Cor. 3.14]).

Using the first paragraph, we can then perform Chinese remaindering for the cost $O(\mathscr{C}(\mathbf{P}))$ of $\mathsf{comb}_{\mathbf{P}}$, plus modular multiplications by the F_i's. The cost of the latter is $O(\mathscr{M}(m))$, by the super-linearity of $\mathscr{M}$, which is $O(\mathscr{C}(\mathbf{P}))$. The same holds for the inverse of $\mathsf{comb}_{\mathbf{P}}$, which is reduced to $\mathsf{red}_{\mathbf{P}}$ and modular multiplications by the F_i's. □

In matrix terms, we will need the following notation in the next sections (here and hereafter, we use canonical monomial bases for vector spaces such as $\mathbb{F}[x]_m$):

- $\mathbb{X}_{\mathbf{P}} \in \mathbb{F}^{m \times m}$ is the matrix of the mapping $\mathsf{comb}_{\mathbf{P}}$;
- $\mathbb{W}_{\mathbf{P}} \in \mathbb{F}^{m \times m}$ is the matrix of the mapping $\mathsf{red}_{\mathbf{P}}$.

LEMMA 4. *Let $\mathbf{P} = P_1, \ldots, P_d$ as above be given and assume that $\mathscr{H}_{\mathbf{P}}$ holds. Then, after a precomputation of time $O(\mathscr{C}(\mathbf{P}) + \mathscr{D}(\mathbf{P}))$, one can compute $\mathbb{X}_{\mathbf{P}}\,\mathsf{u}$, as well as $\mathbb{W}_{\mathbf{P}}\,\mathsf{u}$, $\mathbb{W}_{\mathbf{P}}^t\,\mathsf{u}$, $\mathbb{W}_{\mathbf{P}}^{-1}\,\mathsf{u}$ and $\mathbb{W}_{\mathbf{P}}^{-t}\,\mathsf{u}$ for any u in $\mathbb{F}^m$ in time $O(\mathscr{C}(\mathbf{P}))$.*

Proof. For $\mathbb{X}_{\mathbf{P}}\,\mathsf{u}$, $\mathbb{W}_{\mathbf{P}}\,\mathsf{u}$ and $\mathbb{W}_{\mathbf{P}}^{-1}\,\mathsf{u}$, this is nothing else than the previous lemma. To compute with the transpose of $\mathbb{W}_{\mathbf{P}}$, we could rely on the transposition principle [6], but it is actually possible to give direct algorithms. Algorithm TSimulMod in [5] shows how to reduce the computation of $\mathbb{W}_{\mathbf{P}}^t\,\mathsf{u}$ to an application of $\mathsf{comb}_{\mathbf{P}}$ and a few power series multiplications / inversions that involve only P and the P_i's, and whose costs add up to $O(\mathscr{M}(m))$. Thus, the claimed $O(\mathscr{C}(\mathbf{P}))$ follows from the previous lemma. It is straightforward to invert that algorithm step-by-step, obtaining an algorithm for computing $\mathbb{W}_{\mathbf{P}}^{-t}\,\mathsf{u}$ in time $O(\mathscr{C}(\mathbf{P}))$ as well. □

2.3. Modular arithmetic. Given a monic polynomial $P \in \mathbb{F}[x]$ of degree m, we will frequently work with the residue class ring $\mathbb{A} = \mathbb{F}[x]/P$; doing so, we will identify its elements with polynomials in $\mathbb{F}[x]_m$. In this subsection, we review several questions related to computations in $\mathbb{A}$, such as modular multiplication and its transpose.

As a preamble, we define two useful matrices related to P. First, for $\ell \geqslant 1$, $\mathbb{W}_{P,\ell} \in \mathbb{F}^{m\times\ell}$ denotes the matrix of reduction modulo P, which maps a polynomial $F \in \mathbb{F}[x]_\ell$ to $F \bmod P$. Computationally, applying $\mathbb{W}_{P,\ell}$ to a vector amounts to doing a Euclidean division. Next, writing $P = p_0 + p_1 x + \cdots + p_m x^m$ (so that $p_m = 1$), we will denote by $\mathbb{Y}_P$ the $m \times m$ Hankel matrix

$$\mathbb{Y}_P = \begin{bmatrix} p_1 & \cdots & p_{m-1} & 1 \\ \vdots & \iddots & 1 & \\ p_{m-1} & \iddots & & \\ 1 & & & \end{bmatrix}.$$

Operations with $\mathbb{Y}_P$ are fast, due to the triangular Hankel nature of this matrix. The only non-trivial part of the following lemma concerns the inverse of $\mathbb{Y}_P$. It is proved in [33, §2.5] for triangular Toeplitz matrices; the extension to the Hankel case is straightforward, since $\mathbb{Y}_P^{-1} = (\mathbb{J}_m \mathbb{Y}_P)^{-1}\mathbb{J}_m$ with $\mathbb{J}_m\mathbb{Y}_P$ triangular Toeplitz.

LEMMA 5. *Given P, for u in $\mathbb{F}^m$, one can compute $\mathbb{Y}_P\,\mathsf{u}$ and $\mathbb{Y}_P^{-1}\,\mathsf{u}$ in $O(\mathscr{M}(m))$.*

Let us now discuss multiplication in $\mathbb{A}$, often called *modular multiplication*. Computationally, it is done by a standard polynomial multiplication, followed by a Euclidean division; altogether, this takes time $O(\mathscr{M}(m))$.

In matrix terms, recall that $\mathbb{M}_P$ denotes the $m \times m$ companion matrix associated with P; equivalently, this is the matrix of the multiplication-by-x endomorphism in $\mathbb{A}$. More generally, for F in $\mathbb{F}[x]$, we denote by $\mathbb{M}_{F,P}$ the $m \times m$ matrix of multiplication by F in $\mathbb{A}$; that is, the matrix whose (i, j) entry is the coefficient of x^i in $Fx^j \bmod P$, for $i, j = 0, \ldots, m-1$. Thus, by what was said above, applying $\mathbb{M}_{F,P}$ to a vector can be done in time $O(\mathscr{M}(m))$. Note also that $\mathbb{M}_{x,P} = \mathbb{M}_P$ and that $\mathbb{M}_{F,P} = F(\mathbb{M}_P)$.

Yet more generally, for $\ell \geqslant 1$, $\mathbb{M}_{F,P,\ell} \in \mathbb{F}^{m\times\ell}$ is the matrix whose entries are the coefficients of the same polynomials $Fx^j \bmod P$ as above, but now for $j = 0, \ldots, \ell-1$. The following easy lemma simply says that for F in $\mathbb{F}[x]_m$ and G in $\mathbb{F}[x]_\ell$, $(GF) \bmod P = ((G \bmod P)F) \bmod P$.

LEMMA 6. *Let F be in $\mathbb{F}[x]_m$ and $\ell \geqslant 1$. Then $\mathbb{M}_{F,P,\ell} = \mathbb{M}_{F,P}\,\mathbb{W}_{P,\ell}$.*

We will also use a dual operation, involving the space $\mathbb{A}^*$ of $\mathbb{F}$-linear forms $\mathbb{A} \to \mathbb{F}$. Such a linear form will be given by the values it takes on the monomial basis $1, x, \ldots, x^{m-1}$ of $\mathbb{A}$. Then, *transposed multiplication* is the operation mapping $(F, \lambda) \in \mathbb{A} \times \mathbb{A}^*$ to the linear form $\lambda' = F \circ \lambda$, defined by $\lambda'(x^i) = \lambda(x^i F)$, where the multiplication takes place in $\mathbb{A}$. The name reflects the fact that for fixed F, transposed multiplication by F is indeed the dual map of the multiplication-by-F endomorphism of $\mathbb{A}$. In matrix terms, transposed product by F amounts to multiplication by $\mathbb{M}_{F,P}^t$.

Transposed products can be computed in the same time $O(\mathscr{M}(m))$ as "standard" modular multiplications, by a dual form of modular multiplication [6]. However, such an algorithm relies on middle product techniques [15], which are not straightforward to describe. The following lemma shows an alternative approach, with same cost up to a constant factor: to perform a transposed product by F, it suffices to do a modular multiplication by F, with a pre- and post-multiplication by $\mathbb{Y}_P$ and its inverse (which

can both be done in $O(\mathscr{M}(m))$, see Lemma 5). For $F = x$, this result is sometimes referred to as saying that $\mathbb{Y}_P$ is a *symmetrizer* for the polynomial P; see [22, p. 455].

LEMMA 7. *For all $F \in \mathbb{F}[x]_m$, we have $\mathbb{Y}_P\, \mathbb{M}_{F,P}^t = \mathbb{M}_{F,P}\, \mathbb{Y}_P$. Equivalently, $\mathbb{M}_{F,P}^t = \mathbb{Y}_P^{-1}\, \mathbb{M}_{F,P}\, \mathbb{Y}_P$.*

Proof. By linearity, it is enough to consider the case of $F = x^\ell$, for $0 \leqslant \ell < m$. The case $\ell = 0$ is clear, since $\mathbb{M}_{1,P}$ is the identity matrix. For $\ell = 1$, this result is well-known, see for instance [22, Ex. 3, Ch. 13]; it can be proved by simple inspection. Once the claim is established for x, an easy induction shows that it holds for x^ℓ, for an arbitrary $\ell \geqslant 1$, using the fact that $\mathbb{M}_{x^{\ell+1},P} = \mathbb{M}_P\, \mathbb{M}_{x^\ell,P} = \mathbb{M}_{x^\ell,P}\, \mathbb{M}_P$. ∎

We now discuss Krylov matrices derived from (transposed) companion matrices.

LEMMA 8. *Let $\mathsf{v} \in \mathbb{F}^m$, $\ell \geqslant 1$, and $F = \mathsf{pol}(\mathsf{v}) \in \mathbb{F}[x]_m$. Then*

- $\mathbb{K}(\mathbb{M}_P, \mathsf{v}, \ell) = \mathbb{M}_{F,P}\, \mathbb{W}_{P,\ell} = \mathbb{M}_{F,P,\ell}$ *and*
- $\mathbb{K}(\mathbb{M}_P^t, \mathsf{v}, \ell) = \mathbb{Y}_P^{-1}\, \mathbb{M}_{G,P}\, \mathbb{W}_{P,\ell}$ *with* $G = \mathsf{pol}(\mathbb{Y}_P\, \mathsf{v})$.

Proof. The first assertion is clear. Indeed, the columns of the Krylov matrix $\mathbb{K}(\mathbb{M}_P, \mathsf{v}, \ell)$ are the coefficient vectors of the polynomials $x^j F \bmod P$, for $0 \leqslant j < \ell$, so that $\mathbb{K}(\mathbb{M}_P, \mathsf{v}, \ell) = \mathbb{M}_{F,P,\ell}$; the claim now follows from Lemma 6. For the second assertion, the fact that $\mathbb{M}_P^t = \mathbb{Y}_P^{-1}\, \mathbb{M}_P\, \mathbb{Y}_P$, which follows from Lemma 7, implies that $\mathbb{K}(\mathbb{M}_P^t, \mathsf{v}, \ell) = \mathbb{Y}_P^{-1}\, \mathbb{K}(\mathbb{M}_P, \mathbb{Y}_p\, \mathsf{v}, \ell)$. Using the first point concludes the proof. ∎

2.4. Computations with a family of polynomials. We now consider a family of monic polynomials $\mathbf{P} = P_1, \ldots, P_d$, and briefly discuss the extension of the previous claims to this context. As before, we write $m_i = \deg(P_i)$ and $m = m_1 + \cdots + m_d$.

In terms of matrices, we have already mentioned the definition of the block-diagonal companion matrix $\mathbb{M}_{\mathbf{P}}$ associated with $\mathbf{P}$, whose blocks are $\mathbb{M}_{P_1}, \ldots, \mathbb{M}_{P_d}$. Similarly, $\mathbb{Y}_{\mathbf{P}}$ will denote the block-diagonal matrix with Hankel blocks $\mathbb{Y}_{P_1}, \ldots, \mathbb{Y}_{P_d}$. The next lemma, which is about $\mathbb{Y}_{\mathbf{P}}$ and its inverse, is a direct consequence of Lemma 5; the complexity estimate follows from the super-linearity of the function $\mathscr{M}$.

LEMMA 9. *Given $\mathbf{P}$, for u in $\mathbb{F}^m$, one can compute $\mathbb{Y}_{\mathbf{P}}\, \mathsf{u}$ and $\mathbb{Y}_{\mathbf{P}}^{-1}\, \mathsf{u}$ in $O(\mathscr{M}(m))$.*

Another straightforward result is the following extension of Lemma 7.

LEMMA 10. *The relation $\mathbb{Y}_{\mathbf{P}}\, \mathbb{M}_{\mathbf{P}}^t = \mathbb{M}_{\mathbf{P}}\, \mathbb{Y}_{\mathbf{P}}$ holds.*

Next, we discuss computations with Krylov matrices derived from $\mathbb{M}_{\mathbf{P}}$ and its transpose. We will only need some special cases, for which very simple formulas are available (and for which invertibility will be easy to prove). Recall that $\mathbb{W}_{\mathbf{P}} \in \mathbb{F}^{m \times m}$ denotes the matrix of the map $\mathsf{red}_{\mathbf{P}}$ of multiple reduction modulo $P_1, \ldots, P_d$; thus, it is obtained by stacking the matrices $\mathbb{W}_{P_1,m}, \ldots, \mathbb{W}_{P_d,m}$.

LEMMA 11. *Writing $\mathsf{e}_{m,i}$ to denote the ith unit vector in $\mathbb{F}^m$, the following holds:*

- $\mathbb{K}(\mathbb{M}_{\mathbf{P}}, \mathsf{v}, m) = \mathbb{W}_{\mathbf{P}}$ *with* $\mathsf{v} = [\mathsf{e}_{m_1,1}^t | \cdots | \mathsf{e}_{m_d,1}^t]^t \in \mathbb{F}^m$, *and*
- $\mathbb{K}(\mathbb{M}_{\mathbf{P}}^t, \mathsf{w}, m) = \mathbb{Y}_{\mathbf{P}}^{-1}\, \mathbb{W}_{\mathbf{P}}$ *with* $\mathsf{w} = [\mathsf{e}_{m_1,m_1}^t | \cdots | \mathsf{e}_{m_d,m_d}^t]^t \in \mathbb{F}^m$.

Proof. The block structure of $\mathbb{M}_{\mathbf{P}}$ and the definition of v imply that $\mathbb{K}(\mathbb{M}_{\mathbf{P}}, \mathsf{v}, m)$ is obtained by stacking the matrices $\mathbb{K}(\mathbb{M}_{P_i}, \mathsf{e}_{m_i,1}, m)$ for $i = 1, \ldots, d$. Since $\mathsf{pol}(\mathsf{e}_{m_i,1}) = 1 \in \mathbb{F}[x]_{m_i}$, Lemma 8 gives $\mathbb{K}(\mathbb{M}_{P_i}, \mathsf{e}_{m_i,1}, m) = \mathbb{W}_{P_i,m}$ for all i; this proves the first case. In the second case, $\mathbb{K}(\mathbb{M}_{\mathbf{P}}^t, \mathsf{w}, m)$ is decomposed into blocks $\mathbb{K}(\mathbb{M}_{P_i}^t, \mathsf{e}_{m_i,m_i}, m)$, which, by Lemma 8 and since $\mathsf{pol}(\mathbb{Y}_{P_i} \mathsf{e}_{m_i,m_i}) = 1 \in \mathbb{F}[x]_{m_i}$, are equal to $\mathbb{Y}_{P_i}^{-1}\, \mathbb{W}_{P_i,m}$. ∎

Finally, the following lemma discusses computations involving two families of monic polynomials $\mathbf{P}$ and $\mathbf{Q}$.

LEMMA 12. *Let* $\mathbf{P} = P_1, \ldots, P_d$ *and* $\mathbf{Q} = Q_1, \ldots, Q_e$, *such that* $\mathscr{H}_{\mathbf{P}}$ *and* $\mathscr{H}_{\mathbf{Q}}$ *hold, and let* $P = P_1 \cdots P_d$ *and* $Q = Q_1 \cdots Q_e$. *Then, the following holds:*

- *If* $\gcd(P, Q) = 1$, *we can compute all* $1/Q \bmod P_i$, *for* $i = 1, \ldots, d$, *in time* $O(\mathscr{D}(\mathbf{P}, \mathbf{Q}) + \mathscr{C}(\mathbf{P}, \mathbf{Q}))$.
- *Let* $n = \deg(Q)$ *and* $\widetilde{Q} = \mathsf{rev}(Q, n)$. *If* $\gcd(P, \widetilde{Q}) = 1$, *we can compute all* $1/\widetilde{Q} \bmod P_i$, *for* $i = 1, \ldots, d$, *in time* $O(\mathscr{D}(\mathbf{P}, \mathbf{Q}) + \mathscr{C}(\mathbf{P}, \mathbf{Q}))$.

Proof. Let $p = \max(m, n)$. We first discuss a particular case, where $d = e = 1$ and $P = x^m - \varphi$ and $Q = x^n - \psi$. In this case, we can compute $1/Q \bmod P$ in linear time $O(p)$ [39, Lemma 3], provided it is well-defined; the same holds for $1/\widetilde{Q} \bmod P$. Thus, our claim holds, since in this case, $\mathscr{D}(\mathbf{P}, \mathbf{Q}) = m + n$.

If P is not as above, we compute Q in time $O(\mathscr{C}(\mathbf{Q}) + \mathscr{D}(\mathbf{Q}))$ using Lemma 3, then $Q \bmod P$ in time $O(\mathscr{M}(p))$, which is $O(\mathscr{C}(\mathbf{P}, \mathbf{Q}))$. Using Lemma 3 again, we deduce all $Q \bmod P_i$, for $i \leqslant d$, in time $O(\mathscr{C}(\mathbf{P}))$; finally, we invert all remainders using fast extended GCD with the P_i's in time $O(\sum_{i \leqslant d} \mathscr{M}(m_i)\lg(m_i))$; under our assumption on P, this is $O(\mathscr{D}(\mathbf{P}))$. The total cost is $O(\mathscr{D}(\mathbf{P}, \mathbf{Q}) + \mathscr{C}(\mathbf{P}, \mathbf{Q}))$, so our claim for the inverses of Q modulo the P_i's holds. We proceed similarly for $\widetilde{Q}$.

The last case to consider is when $d = 1$ and $P = x^m - \varphi$, but with Q not of the form $x^n - \psi$. We proceed in the converse direction: we first reduce and invert P modulo all Q_j, for all $j \leqslant e$; this takes time $O(\mathscr{D}(\mathbf{P}, \mathbf{Q}) + \mathscr{C}(\mathbf{P}, \mathbf{Q}))$. By Chinese Remaindering, we obtain $R = 1/P \bmod Q$, without increasing the cost. Knowing R, we deduce $S = 1/Q \bmod P$, since R and S satisfy $RP + SQ = 1$; this costs an extra $O(\mathscr{M}(p))$, which is $O(\mathscr{C}(\mathbf{P}, \mathbf{Q}))$, so our claim is proved. We proceed similarly for $\widetilde{Q}$. ∎

3. Inverting displacement operators. Let us consider $\mathbf{P} = P_1, \ldots, P_d$ and $\mathbf{Q} = Q_1, \ldots, Q_e$ tuples of monic polynomials in $\mathbb{F}[x]$, and let $m_1, \ldots, m_d$, $n_1, \ldots, n_e$, m, n, P and Q be as before. We assume the coprimality conditions $\mathscr{H}_{\mathbf{P}}$ and $\mathscr{H}_{\mathbf{Q}}$. In this section, we establish inversion formulas for the two basic operators of Sylvester and Stein types associated with $(\mathbf{P}, \mathbf{Q})$. In other words, given two matrices (G, H) in $\mathbb{F}^{m \times \alpha} \times \mathbb{F}^{n \times \alpha}$, we show how to reconstruct the matrices A and A' in $\mathbb{F}^{m \times n}$ such that

$$\nabla_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}_{\mathbf{Q}}^t}(\mathsf{A}) = \mathsf{G}\mathsf{H}^t \quad \text{and} \quad \Delta_{\mathbb{M}_{\mathbf{P}}, \mathbb{M}_{\mathbf{Q}}^t}(\mathsf{A}') = \mathsf{G}\mathsf{H}^t,$$

provided the corresponding operators are invertible. We will use the following objects.

- We denote by $\mathsf{g}_1, \ldots, \mathsf{g}_\alpha$ and $\mathsf{h}_1, \ldots, \mathsf{h}_\alpha$ the columns of G and H. Corresponding to the partition of $\mathbb{M}_{\mathbf{P}}$ into blocks, we partition each g_k, $k \leqslant \alpha$, into smaller column vectors $\mathsf{g}_{1,k}, \ldots, \mathsf{g}_{d,k}$; similarly, we partition each h_k into $\mathsf{h}_{1,k}, \ldots, \mathsf{h}_{e,k}$ according to the block structure of $\mathbb{M}_{\mathbf{Q}}^t$. The matrices G and H themselves are partitioned into matrices $\mathsf{G}_1, \ldots, \mathsf{G}_d$ and $\mathsf{H}_1, \ldots, \mathsf{H}_e$; G_i has size $m_i \times \alpha$ and columns $\mathsf{g}_{i,1}, \ldots, \mathsf{g}_{i,\alpha}$, whereas H_j has size $n_j \times \alpha$ and columns $\mathsf{h}_{j,1}, \ldots, \mathsf{h}_{j,\alpha}$.
- For $i = 1, \ldots, d$ and $k = 1, \ldots, \alpha$, we let $g_{i,k} = \mathsf{pol}(\mathsf{g}_{i,k})$, so that $g_{i,k}$ is in $\mathbb{F}[x]_{m_i}$. Similarly, for $j = 1, \ldots, e$, we define $h_{j,k} = \mathsf{pol}(\mathsf{h}_{j,k}) \in \mathbb{F}[x]_{n_j}$.
  We further let γ_k be the unique polynomial in $\mathbb{F}[x]_m$ such that $\gamma_k \bmod P_i = g_{i,k}$ holds for $i = 1, \ldots, d$; in other words, we have

$$\gamma_k = \mathsf{crt}_{\mathbf{P}}(g_{1,k}, \ldots, g_{d,k}).$$

The polynomial η_k is defined similarly, replacing $g_{i,k}$ and P_i by $h_{j,k}$ and Q_j.

In the following theorem, recall that the necessary and sufficient condition for the Sylvester operator to be invertible is that $\gcd(P, Q) = 1$; for the Stein operator, the condition is that $\gcd(P, \widetilde{Q}) = 1$, with $\widetilde{Q} = \mathsf{rev}(Q, n)$.

THEOREM 13. *The following holds:*

- *Suppose that* $\gcd(P, Q) = 1$. *Then, the unique matrix* $\mathsf{A} \in \mathbb{F}^{m\times n}$ *such that* $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}(\mathsf{A}) = \mathsf{G}\mathsf{H}^t$ *is given by*

$$\mathsf{A} = \mathbb{V}_{\mathbf{P},\mathbf{Q}}\, \mathbb{W}_{\mathbf{P}} \left(\sum_{k\leqslant\alpha} \mathbb{M}_{\gamma_k,P,n}\, \mathbb{M}_{\eta_k,Q} \right) \mathbb{X}_{\mathbf{Q}}\, \mathbb{Y}_{\mathbf{Q}},$$

  where $\mathbb{V}_{\mathbf{P},\mathbf{Q}} \in \mathbb{F}^{m\times m}$ *is the block-diagonal matrix with blocks* $\mathbb{M}_{Q^{-1},P_i}$ *for* $i = 1, \ldots, d$, *where* $\mathbb{M}_{Q^{-1},P_i}$ *denotes* $\mathbb{M}_{Q^{-1} \bmod P_i,P_i}$.
- *Suppose that* $\gcd(P, \widetilde{Q}) = 1$. *Then, the unique matrix* $\mathsf{A}' \in \mathbb{F}^{m\times n}$ *such that* $\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}(\mathsf{A}') = \mathsf{G}\mathsf{H}^t$ *is given by*

$$\mathsf{A}' = \mathbb{V}'_{\mathbf{P},\mathbf{Q}}\, \mathbb{W}_{\mathbf{P}} \left(\sum_{k\leqslant\alpha} \mathbb{M}_{\gamma_k,P,n}\, \mathbb{J}_n\, \mathbb{M}_{\eta_k,Q} \right) \mathbb{X}_{\mathbf{Q}}\, \mathbb{Y}_{\mathbf{Q}},$$

  where $\mathbb{V}'_{\mathbf{P},\mathbf{Q}} \in \mathbb{F}^{m\times m}$ *is the block-diagonal matrix with blocks* $\mathbb{M}_{\widetilde{Q}^{-1},P_i}$ *for* $i = 1, \ldots, d$, *where* $\mathbb{M}_{\widetilde{Q}^{-1},P_i}$ *denotes* $\mathbb{M}_{\widetilde{Q}^{-1} \bmod P_i,P_i}$.

The following corollary on the cost of matrix-vector multiplication follows directly; the more difficult case of matrix-matrix multiplication will be handled in Section 5.

COROLLARY 14. *We can take*

$$\mathscr{C}_{\mathrm{mul}}(\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}, \alpha, 1) = O(\mathscr{D}(\mathbf{P}, \mathbf{Q}) + \alpha\mathscr{C}(\mathbf{P}, \mathbf{Q}))$$

and

$$\mathscr{C}_{\mathrm{mul}}(\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}, \alpha, 1) = O(\mathscr{D}(\mathbf{P}, \mathbf{Q}) + \alpha\mathscr{C}(\mathbf{P}, \mathbf{Q})).$$

Proof. The proof is mostly the same in both cases; it amounts to estimating first the cost of computing the polynomials that define the matrices involved in Theorem 13, and then the cost of multiplying each of these matrices by a single vector.

Given the families of polynomials $\mathbf{P}$ and $\mathbf{Q}$, we can compute the products P and Q in time $O(\mathscr{C}(\mathbf{P}, \mathbf{Q}) + \mathscr{D}(\mathbf{P}, \mathbf{Q}))$, by Lemma 3. Moreover, since both assumptions $\mathscr{H}_{\mathbf{P}}$ and $\mathscr{H}_{\mathbf{Q}}$ hold and depending on whether $\gcd(P, Q) = 1$ or $\gcd(P, \widetilde{Q}) = 1$, we deduce from Lemma 12 that we can compute the inverses of Q, or of $\widetilde{Q}$, modulo all P_i's in time $O(\mathscr{D}(\mathbf{P}, \mathbf{Q})+\mathscr{C}(\mathbf{P}, \mathbf{Q}))$. Finally, given further the generator (G, H) in $\mathbb{F}^{m\times\alpha} \times \mathbb{F}^{n\times\alpha}$, it follows from Lemma 3 that the polynomials $\gamma_1, \ldots, \gamma_\alpha$ and $\eta_1, \ldots, \eta_\alpha$ can be obtained in time $O(\mathscr{D}(\mathbf{P})+\alpha\mathscr{C}(\mathbf{P}))$ and $O(\mathscr{D}(\mathbf{Q})+\alpha\mathscr{C}(\mathbf{Q}))$, respectively. Thus, the overall cost of deducing the needed polynomials from $\mathbf{P}$, $\mathbf{Q}$, G, H is in $O(\mathscr{D}(\mathbf{P}, \mathbf{Q}) + \alpha\mathscr{C}(\mathbf{P}, \mathbf{Q}))$.

We now turn to the cost of the matrix-vector products performed when applying Theorem 13 to the evaluation of $\mathsf{A}\mathsf{u}$ or $\mathsf{A}'\mathsf{u}$ for some u in $\mathbb{F}^n$. Lemma 9 shows that the cost of multiplying $\mathbb{Y}_{\mathbf{Q}}$ by u is $O(\mathscr{M}(n))$, which is $O(\mathscr{M}(p))$. By Lemma 4, the cost for $\mathbb{X}_{\mathbf{Q}}$ is $O(\mathscr{D}(\mathbf{Q}) + \mathscr{C}(\mathbf{Q}))$. The inner sum amounts to $O(\alpha)$ multiplications modulo P or Q, for a total of $O(\alpha\mathscr{M}(p))$. Lemma 4 also shows that the cost for $\mathbb{W}_{\mathbf{P}}$ is $O(\mathscr{D}(\mathbf{P}) + \mathscr{C}(\mathbf{P}))$. Finally, multiplying by $\mathbb{V}_{\mathbf{P},\mathbf{Q}}$ or $\mathbb{V}'_{\mathbf{P},\mathbf{Q}}$ amounts to performing modular multiplications modulo all P_i's, and this can be done in time $O(\mathscr{M}(m)) \subset O(\mathscr{M}(p))$. Hence the cost of all these matrix-vector products is in $O(\mathscr{D}(\mathbf{P}, \mathbf{Q}) + \mathscr{C}(\mathbf{P}, \mathbf{Q}) + \alpha\mathscr{M}(p))$; this fits the announced bound, since $\mathscr{M}(p) \leqslant \mathscr{C}(\mathbf{P}, \mathbf{Q})$. ∎

The rest of this section is devoted to proving Theorem 13 above.

3.1. Sylvester operator $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}$. Assuming that $\gcd(P,Q)=1$, we show how to reconstruct the unique matrix $\mathsf{A}\in\mathbb{F}^{m\times n}$ such that $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}(\mathsf{A})=\mathsf{G}\mathsf{H}^t$. We first show how to reconstruct all blocks in an ad-hoc block decomposition of A, then use Chinese remaindering.

Step 1: reconstruction of the blocks of A. Partitioning the matrix A into blocks $\mathsf{A}_{i,j}$ conformally with the block-diagonal structure of the matrices $\mathbb{M}_{\mathbf{P}}$ and $\mathbb{M}_{\mathbf{Q}}$, we have $\nabla_{\mathbb{M}_{P_i},\mathbb{M}^t_{Q_j}}(\mathsf{A}_{i,j})=\mathsf{G}_i\mathsf{H}_j^t$ for all i,j. In this paragraph, we prove a reconstruction formula for each block $\mathsf{A}_{i,j}$.

LEMMA 15. *For all $i\leqslant d$ and $j\leqslant e$, we have*

$$\mathsf{A}_{i,j}=\sum_{k\leqslant\alpha}\mathbb{M}_{g_{i,k},P_i}\,\mathbb{M}_{Q_j^{-1},P_i,n_j}\,\mathbb{M}_{h_{j,k},Q_j}\,\mathbb{Y}_{Q_j}.$$

Proof. Fix i and j. Note that $\mathbb{M}_{P_i}$ and $\mathbb{M}_{Q_j}$ cannot be simultaneously singular, since P_i and Q_j are coprime. Then, for all $\ell\geqslant 1$, a slight variation of [35, Theorem 4.8] (with both cases "$\mathbb{M}_{P_i}$ nonsingular" and "$\mathbb{M}_{Q_j}$ nonsingular" covered by a single identity) gives

$$\mathbb{M}^\ell_{P_i}\,\mathsf{A}_{i,j}-\mathsf{A}_{i,j}\,(\mathbb{M}^t_{Q_j})^\ell=\sum_{k\leqslant\alpha}\mathbb{K}(\mathbb{M}_{P_i},\mathsf{g}_{i,k},\ell)\,\mathbb{J}_\ell\,\mathbb{K}(\mathbb{M}_{Q_j},\mathsf{h}_{j,k},\ell)^t,$$

where $\mathbb{K}(\mathbb{M}_{P_i},\mathsf{g}_{i,k},\ell)\in\mathbb{F}^{m_i\times\ell}$ denotes the Krylov matrix of column length ℓ associated with the matrix $\mathbb{M}_{P_i}$ and the vector $\mathsf{g}_{i,k}$.

Writing $Q_j=q_{j,0}+\cdots+q_{j,n_j}x^{n_j}$, we obtain, for $1\leqslant\ell\leqslant n_j$,

$$\begin{aligned}q_{j,\ell}\,\mathbb{M}^\ell_{P_i}\,\mathsf{A}_{i,j}-\mathsf{A}_{i,j}\,q_{j,\ell}\,(\mathbb{M}^t_{Q_j})^\ell&=\sum_{k\leqslant\alpha}\mathbb{K}(\mathbb{M}_{P_i},\mathsf{g}_{i,k},\ell)\,q_{j,\ell}\,\mathbb{J}_\ell\,\mathbb{K}(\mathbb{M}_{Q_j},\mathsf{h}_{j,k},\ell)^t\\&=\sum_{k\leqslant\alpha}\mathbb{K}(\mathbb{M}_{P_i},\mathsf{g}_{i,k},n_j)\,q_{j,\ell}\,\mathbb{J}_{\ell,n_j}\,\mathbb{K}(\mathbb{M}_{Q_j},\mathsf{h}_{j,k},n_j)^t,\end{aligned}$$

since we can inflate all matrices $\mathbb{J}_\ell$ to size $n_j\times n_j$, replacing them by $\mathbb{J}_{\ell,n_j}$. Note that the above equality then also holds for $\ell=0$, since $\mathbb{J}_{0,n_j}=0$ by convention. Summing over all $\ell=0,\ldots,n_j$ and using the fact that $Q_j(\mathbb{M}^t_{Q_j})=Q_j(\mathbb{M}_{Q_j})^t=0$, we deduce

$$Q_j(\mathbb{M}_{P_i})\,\mathsf{A}_{i,j}=\sum_{k\leqslant\alpha}\mathbb{K}(\mathbb{M}_{P_i},\mathsf{g}_{i,k},n_j)\,\mathbb{Y}_{Q_j}\,\mathbb{K}(\mathbb{M}_{Q_j},\mathsf{h}_{j,k},n_j)^t.$$

Using the first part of Lemma 8 together with the fact that for $\gcd(P_i,Q_j)=1$ the matrix $Q_j(\mathbb{M}_{P_i})=\mathbb{M}_{Q_j,P_i}$ is invertible with inverse $\mathbb{M}^{-1}_{Q_j,P_i}=\mathbb{M}_{Q_j^{-1},P_i}$, we obtain

$$\mathsf{A}_{i,j}=\sum_{k\leqslant\alpha}\mathbb{M}_{Q_j^{-1},P_i}\,\mathbb{M}_{g_{i,k},P_i,n_j}\,\mathbb{Y}_{Q_j}\,\mathbb{M}^t_{h_{j,k},Q_j}.$$

Hence, applying Lemma 7 to $h_{j,k}$ and since multiplication matrices mod P_i commute,

$$\mathsf{A}_{i,j}=\sum_{k\leqslant\alpha}\mathbb{M}_{g_{i,k},P_i}\,\mathbb{M}_{Q_j^{-1},P_i,n_j}\,\mathbb{M}_{h_{j,k},Q_j}\,\mathbb{Y}_{Q_j}.$$

∎

Step 2: a first reconstruction formula for A. Let $\mathbb{B}_{\mathbf{P},\mathbf{Q}}$ be the $m \times n$ block matrix

$$\mathbb{B}_{\mathbf{P},\mathbf{Q}} = \left[\mathbb{M}_{Q_j^{-1},P_i,n_j}\right]_{\substack{1\leqslant i\leqslant d\\ 1\leqslant j\leqslant e}}.$$

Similarly, for $k \leqslant \alpha$, we define C_k and D_k as the block diagonal matrices, with respectively the multiplication matrices $(\mathbb{M}_{g_{i,k},P_i})_{i\leqslant d}$ and $(\mathbb{M}_{h_{j,k},Q_j})_{j\leqslant e}$ on the diagonal.

Putting all blocks $\mathsf{A}_{i,j}$ together, we deduce the following reconstruction formula for A; the proof is a straightforward application of the previous lemma.

LEMMA 16. *We have*

$$\mathsf{A} = \left(\sum_{k\leqslant\alpha} \mathsf{C}_k\, \mathbb{B}_{\mathbf{P},\mathbf{Q}}\, \mathsf{D}_k\right) \mathbb{Y}_{\mathbf{Q}}.$$

Step 3: using a factorization of $\mathbb{B}_{\mathbf{P},\mathbf{Q}}$. Next, we use the fact that $\mathbb{B}_{\mathbf{P},\mathbf{Q}}$ has a block structure similar to a Cauchy matrix to factor it explicitly. For this, we introduce a matrix related to $\mathbb{W}_{\mathbf{P}}$: we let $\mathbb{W}_{\mathbf{P},n}$ be the $m \times n$ matrix of multiple reduction modulo $P_1, \ldots, P_d$, taking as input a polynomial of degree less than n, instead of m for $\mathbb{W}_{\mathbf{P}}$.

LEMMA 17. *The equality* $\mathbb{B}_{\mathbf{P},\mathbf{Q}} = \mathbb{V}_{\mathbf{P},\mathbf{Q}}\, \mathbb{W}_{\mathbf{P},n}\, \mathbb{X}_{\mathbf{Q}}$ *holds, where* $\mathbb{V}_{\mathbf{P},\mathbf{Q}} \in \mathbb{F}^{m\times m}$ *is the block-diagonal matrix with blocks* $\mathbb{M}_{Q^{-1},P_i}$*, for* $i = 1, \ldots, d$.

Proof. Observe that, given the coefficient vectors of polynomials $F_1, \ldots, F_e$, with $F_j \in \mathbb{F}[x]_{n_j}$ for all j, the matrix $\mathbb{B}_{\mathbf{P},\mathbf{Q}}$ returns the coefficients of

$$\begin{aligned} G_i &= \frac{F_1}{Q_1} + \cdots + \frac{F_e}{Q_e} \bmod P_i \\ &= \frac{F_1 Q_2 \cdots Q_e + \cdots + Q_1 \cdots Q_{e-1} F_e}{Q_1 \cdots Q_e} \bmod P_i, \end{aligned}$$

for $i = 1, \ldots, d$. Computing the polynomials G_i can be done as follows:

- compute $H = F_1 Q_2 \cdots Q_e + \cdots + Q_1 \cdots Q_{e-1} F_e$ by calling $\mathsf{comb}_{\mathbf{P}}$;
- compute $H_i = H \bmod P_i$, for $i = 1, \ldots, d$;
- deduce $G_i = H_i/Q \bmod P_i$, for $i = 1, \ldots, d$.

This proves the requested factorization of $\mathbb{B}_{\mathbf{P},\mathbf{Q}}$. ∎

As a result, we deduce the following equalities for A:

$$\begin{aligned} \mathsf{A} &= \left(\sum_{k\leqslant\alpha} \mathsf{C}_k\, \mathbb{V}_{\mathbf{P},\mathbf{Q}}\, \mathbb{W}_{\mathbf{P},n}\, \mathbb{X}_{\mathbf{Q}}\, \mathsf{D}_k\right) \mathbb{Y}_{\mathbf{Q}} \\ &= \mathbb{V}_{\mathbf{P},\mathbf{Q}} \left(\sum_{k\leqslant\alpha} \mathsf{C}_k\, \mathbb{W}_{\mathbf{P},n}\, \mathbb{X}_{\mathbf{Q}}\, \mathsf{D}_k\right) \mathbb{Y}_{\mathbf{Q}}. \end{aligned}$$

The last equality is due to the fact that the block-diagonal matrices C_k and $\mathbb{V}_{\mathbf{P},\mathbf{Q}}$ commute (since all pairwise corresponding blocks are multiplication matrices modulo the same polynomials P_i).

Step 4: using Chinese remaindering. The next step will allow us to take $\mathbb{W}_{\mathbf{P},n}$ and $\mathbb{X}_{\mathbf{P}}$ out of the inner sum. For any polynomial H, any $i \leqslant d$ and any $k \leqslant \alpha$, it is equivalent to *(i)* reduce H modulo P_i and multiply it by $g_{i,k}$ modulo P_i and *(ii)* multiply H by the polynomial γ_k (defined above) modulo P, and reduce it modulo P_i. In other words, we have the commutation relation $\mathsf{C}_k\, \mathbb{W}_{\mathbf{P},n} = \mathbb{W}_{\mathbf{P}}\, \mathbb{M}_{\gamma_k,P,n}$. Similarly, $\mathbb{X}_{\mathbf{Q}}\, \mathsf{D}_k = \mathbb{M}_{\eta_k,Q}\, \mathbb{X}_{\mathbf{Q}}$ and this concludes the proof of the first part of Theorem 13.

3.2. Stein operator $\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}$. The proof for Stein operator case follows the same steps as for the Sylvester case. Let A' be such that $\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}(\mathsf{A}') = \mathsf{G}\mathsf{H}^t$. Just like we decomposed A into blocks $\mathsf{A}_{i,j}$, we decompose A' into blocks $\mathsf{A}'_{i,j}$, such that $\Delta_{\mathbb{M}_{P_i},\mathbb{M}^t_{Q_j}}(\mathsf{A}'_{i,j}) = \mathsf{G}_i\mathsf{H}_j^t$. Extending the notation used above, we write $\widetilde{Q_j} = \mathsf{rev}(Q_j, n_j)$, so that we have $\widetilde{Q} = \widetilde{Q_1}\cdots\widetilde{Q_e}$. The following lemma is then the analogue of Lemma 15 for Stein operators and, as detailed in Appendix B, can be proved in the same way using [35, Theorem 4.7].

LEMMA 18. *For all $i \leqslant d$ and $j \leqslant e$, we have*

$$\mathsf{A}'_{i,j} = \sum_{k\leqslant\alpha} \mathbb{M}_{g_{i,k},P_i}\,\mathbb{M}_{\widetilde{Q_j}^{-1},P_i,n_j}\,\mathbb{J}_{n_j}\,\mathbb{M}_{h_{j,k},Q_j}\,\mathbb{Y}_{Q_j}.$$

Mimicking the construction in the previous section, we introduce the $m\times n$ block matrix $\mathbb{B}'_{\mathbf{P},\mathbf{Q}}$ given by

$$\mathbb{B}'_{\mathbf{P},\mathbf{Q}} = \left[\mathbb{M}_{\widetilde{Q_j}^{-1},P_i,n_j}\right]_{\substack{1\leqslant i\leqslant d\\ 1\leqslant j\leqslant e}}.$$

We will use again the matrices C_k and D_k introduced before, as well as the block-diagonal matrix $\mathbb{D}(\mathbb{J}_{n_j})$ having blocks $\mathbb{J}_{n_j}$ on the diagonal. This leads us to the following analogue of Lemma 16, whose proof is straightforward.

LEMMA 19. *We have*

$$\mathsf{A}' = \left(\sum_{k\leqslant\alpha} \mathsf{C}_k\,\mathbb{B}'_{\mathbf{P},\mathbf{Q}}\,\mathbb{D}(\mathbb{J}_{n_j})\,\mathsf{D}_k\right)\mathbb{Y}_{\mathbf{Q}}.$$

The next step is to use the following factorization of $\mathbb{B}'_{\mathbf{P},\mathbf{Q}}$, or more precisely of $\mathbb{B}'_{\mathbf{P},\mathbf{Q}}\,\mathbb{D}(\mathbb{J}_{n_j})$. The proof is the same as that of Lemma 17, up to taking into account the reversals induced by the matrices $\mathbb{J}_{n_j}$.

LEMMA 20. *The equality $\mathbb{B}'_{\mathbf{P},\mathbf{Q}}\,\mathbb{D}(\mathbb{J}_{n_j}) = \mathbb{V}'_{\mathbf{P},\mathbf{Q}}\,\mathbb{W}_{\mathbf{P},n}\,\mathbb{J}_n\,\mathbb{X}_{\mathbf{Q}}$ holds, where $\mathbb{V}'_{\mathbf{P},\mathbf{Q}} \in \mathbb{F}^{m\times m}$ is the block-diagonal matrix with blocks $\mathbb{M}_{\widetilde{Q}^{-1},P_i}$, for $i = 1,\ldots,d$.*

We conclude the proof of our theorem as before, using the relations

$$\begin{aligned}
\mathsf{C}_k\,\mathbb{V}'_{\mathbf{P},\mathbf{Q}} &= \mathbb{V}'_{\mathbf{P},\mathbf{Q}}\,\mathsf{C}_k,\\
\mathsf{C}_k\,\mathbb{W}_{\mathbf{P},n} &= \mathbb{W}_{\mathbf{P}}\,\mathbb{M}_{\gamma_k,P,n},\\
\mathbb{X}_{\mathbf{Q}}\,\mathsf{D}_k &= \mathbb{M}_{\eta_k,Q}\,\mathbb{X}_{\mathbf{Q}}.
\end{aligned}$$

4. Using operator equivalences. Let $\mathbf{P},\mathbf{Q}$ be as in Theorems 1 and 2, and let as before $p = \max(m,n)$. We are now going to extend the complexity estimates for matrix-vector multiplication given in Corollary 14 to more operators (not only the basic ones), by providing reductions to the Hankel operator $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}$.

THEOREM 21. *Suppose that $\mathscr{H}_{\mathbf{P}}$ and $\mathscr{H}_{\mathbf{Q}}$ hold. Then for any displacement operator $\mathcal{L}$ associated with $(\mathbf{P},\mathbf{Q})$, we can take*

$$\begin{aligned}
\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,\beta) &\leqslant \mathscr{C}_{\mathrm{mul}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}},\alpha+2,\beta) + O\big(\mathscr{D}(\mathbf{P},\mathbf{Q}) + (\alpha+\beta)\mathscr{C}(\mathbf{P},\mathbf{Q})\big),\\
\mathscr{C}_{\mathrm{solve}}(\mathcal{L},\alpha) &\leqslant \mathscr{C}_{\mathrm{solve}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}},\alpha+2) + O\big(\mathscr{D}(\mathbf{P},\mathbf{Q}) + \alpha\mathscr{C}(\mathbf{P},\mathbf{Q})\big),\\
\mathscr{C}_{\mathrm{inv}}(\mathcal{L},\alpha) &\leqslant \mathscr{C}_{\mathrm{inv}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{m,1}},\alpha+2) + O\big(\mathscr{D}(\mathbf{P},\mathbf{Q}) + \alpha\mathscr{C}(\mathbf{P},\mathbf{Q}) + \alpha^{\omega-1}m\big).
\end{aligned}$$

The rest of this section is devoted to the proof of this theorem. In what follows, we write several formulas involving some matrices G and H. In these formulas, G and H will always be taken in $\mathbb{F}^{m\times\alpha}$ and $\mathbb{F}^{n\times\alpha}$, respectively, for some $\alpha \geqslant 1$ (and with $n = m$ when dealing with matrix inversion).

4.1. Reduction to basic operators. Table 4 shows that if a matrix A is structured for one of the operators associated with $(\mathbf{P}, \mathbf{Q})$, then simple pre- and post-multiplications transform A into a matrix which is structured for one of the two *basic* operators associated with $(\mathbf{P}, \mathbf{Q})$. All equivalences in this table follow directly from the identities $\mathbb{Y}_{\mathbf{P}}\,\mathbb{M}_{\mathbf{P}}^t = \mathbb{M}_{\mathbf{P}}\,\mathbb{Y}_{\mathbf{P}}$ and $\mathbb{Y}_{\mathbf{Q}}\,\mathbb{M}_{\mathbf{Q}}^t = \mathbb{M}_{\mathbf{Q}}\,\mathbb{Y}_{\mathbf{Q}}$ (Lemma 10) and from $\mathbb{Y}_{\mathbf{P}}$ and $\mathbb{Y}_{\mathbf{Q}}$ being invertible and symmetric. Combining these equivalences with Lemma 9 will lead to the cost bounds in Lemma 22 below, expressed in terms of basic operators.

TABLE 4
Reduction to the basic operators $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}$ *and* $\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}$.

$$
\begin{aligned}
\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}}(\mathsf{A}) = \mathsf{G}\,\mathsf{H}^t &\iff \nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}(\mathsf{A}\,\mathbb{Y}_{\mathbf{Q}}) = \mathsf{G}\,(\mathbb{Y}_{\mathbf{Q}}\mathsf{H})^t\\
\nabla_{\mathbb{M}_{\mathbf{P}}^t,\mathbb{M}_{\mathbf{Q}}^t}(\mathsf{A}) = \mathsf{G}\,\mathsf{H}^t &\iff \nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}(\mathbb{Y}_{\mathbf{P}}\,\mathsf{A}) = (\mathbb{Y}_{\mathbf{P}}\mathsf{G})\,\mathsf{H}^t\\
\nabla_{\mathbb{M}_{\mathbf{P}}^t,\mathbb{M}_{\mathbf{Q}}}(\mathsf{A}) = \mathsf{G}\,\mathsf{H}^t &\iff \nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}(\mathbb{Y}_{\mathbf{P}}\,\mathsf{A}\,\mathbb{Y}_{\mathbf{Q}}) = (\mathbb{Y}_{\mathbf{P}}\mathsf{G})(\mathbb{Y}_{\mathbf{Q}}\mathsf{H})^t
\end{aligned}
$$

$$
\begin{aligned}
\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}}(\mathsf{A}) = \mathsf{G}\,\mathsf{H}^t &\iff \Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}(\mathsf{A}\,\mathbb{Y}_{\mathbf{Q}}) = \mathsf{G}\,(\mathbb{Y}_{\mathbf{Q}}\mathsf{H})^t\\
\Delta_{\mathbb{M}_{\mathbf{P}}^t,\mathbb{M}_{\mathbf{Q}}^t}(\mathsf{A}) = \mathsf{G}\,\mathsf{H}^t &\iff \Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}(\mathbb{Y}_{\mathbf{P}}\,\mathsf{A}) = (\mathbb{Y}_{\mathbf{P}}\mathsf{G})\,\mathsf{H}^t\\
\Delta_{\mathbb{M}_{\mathbf{P}}^t,\mathbb{M}_{\mathbf{Q}}}(\mathsf{A}) = \mathsf{G}\,\mathsf{H}^t &\iff \Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}(\mathbb{Y}_{\mathbf{P}}\,\mathsf{A}\,\mathbb{Y}_{\mathbf{Q}}) = (\mathbb{Y}_{\mathbf{P}}\mathsf{G})(\mathbb{Y}_{\mathbf{Q}}\mathsf{H})^t.
\end{aligned}
$$

LEMMA 22. *Let $\mathcal{L}$ be a displacement operator associated with $(\mathbf{P}, \mathbf{Q})$. Then:*

- *If $\mathcal{L}$ is a Sylvester operator, we can take*

$$
\begin{aligned}
\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,\beta) &\leqslant \mathscr{C}_{\mathrm{mul}}(\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t},\alpha,\beta) + O((\alpha+\beta)\mathscr{M}(p)),\\
\mathscr{C}_{\mathrm{solve}}(\mathcal{L},\alpha) &\leqslant \mathscr{C}_{\mathrm{solve}}(\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t},\alpha) + O(\alpha\mathscr{M}(p)),\\
\mathscr{C}_{\mathrm{inv}}(\mathcal{L},\alpha) &\leqslant \mathscr{C}_{\mathrm{inv}}(\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t},\alpha) + O(\alpha\mathscr{M}(m));
\end{aligned}
$$

- *If $\mathcal{L}$ is a Stein operator, we can take*

$$
\begin{aligned}
\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,\beta) &\leqslant \mathscr{C}_{\mathrm{mul}}(\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t},\alpha,\beta) + O((\alpha+\beta)\mathscr{M}(p)),\\
\mathscr{C}_{\mathrm{solve}}(\mathcal{L},\alpha) &\leqslant \mathscr{C}_{\mathrm{solve}}(\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t},\alpha) + O(\alpha\mathscr{M}(p)),\\
\mathscr{C}_{\mathrm{inv}}(\mathcal{L},\alpha) &\leqslant \mathscr{C}_{\mathrm{inv}}(\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t},\alpha) + O(\alpha\mathscr{M}(m)).
\end{aligned}
$$

Proof. We give the proof for the Sylvester operator $\mathcal{L} = \nabla_{\mathbb{M}_{\mathbf{P}}^t,\mathbb{M}_{\mathbf{Q}}}$, which appears in the third row of Table 4; the other five cases in that table can be handled similarly.

Suppose first that, given an $\mathcal{L}$-generator (G,H) for $\mathsf{A}\in\mathbb{F}^{m\times n}$ as well as a matrix $\mathsf{B}\in\mathbb{F}^{n\times\beta}$, we want to compute AB. We begin by computing a $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}(\mathsf{A}')$-generator $(\mathbb{Y}_{\mathbf{P}}\mathsf{G},\mathbb{Y}_{\mathbf{Q}}\mathsf{H})$ for $\mathsf{A}' = \mathbb{Y}_{\mathbf{P}}\,\mathsf{A}\,\mathbb{Y}_{\mathbf{Q}}$ (cf. the third row of Table 4). Lemma 9 implies that this takes time $O(\alpha\mathscr{M}(p))$. Then, we evaluate $\mathsf{AB} = \mathbb{Y}_{\mathbf{P}}^{-1}\,\mathsf{A}'\,\mathbb{Y}_{\mathbf{Q}}^{-1}\,\mathsf{B}$ from right to left. The products by $\mathbb{Y}_{\mathbf{P}}^{-1}$ and $\mathbb{Y}_{\mathbf{Q}}^{-1}$ take time $O(\beta\mathscr{M}(p))$, and the product by A' takes time $\mathscr{C}_{\mathrm{mul}}(\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t},\alpha,\beta)$; thus, the first claim is proved.

Suppose now that we are given a vector $\mathsf{b} \in \mathbb{K}^m$ in addition to the matrix A. In order to solve the system $\mathsf{Ax} = \mathsf{b}$, we solve $\mathsf{A}'\mathsf{x}' = \mathsf{b}'$, with A' as defined above and $\mathsf{b}' = \mathbb{Y}_{\mathbf{P}}\mathsf{b}$. The latter system admits a solution if and only if the former one does; if x' is a solution of $\mathsf{A}'\mathsf{x}' = \mathsf{b}'$, then $\mathsf{x} = \mathbb{Y}_{\mathbf{Q}}\mathsf{x}'$ is a solution of $\mathsf{Ax} = \mathsf{b}$. Hence, as before, we set up an $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}$-generator $(\mathbb{Y}_{\mathbf{P}}\mathsf{G}, \mathbb{Y}_{\mathbf{Q}}\mathsf{H})$ for A', using $O(\alpha\mathscr{M}(p))$ operations in $\mathbb{F}$, and compute b' in time $O(\mathscr{M}(m))$. Then, in time $\mathscr{C}_{\text{solve}}(\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}, \alpha)$ we either assert that the system $\mathsf{A}'\mathsf{x}' = \mathsf{b}'$ has no solution, or find such a solution x'. Finally, we recover x in time $O(\mathscr{M}(n))$. This proves the second claim.

Finally, assume that $m = n$ and consider the question of inverting A. This matrix is invertible if and only if the matrix A' defined above is invertible. Again, we set up a $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}$-generator $(\mathbb{Y}_{\mathbf{P}}\mathsf{G}, \mathbb{Y}_{\mathbf{Q}}\mathsf{H})$ for A', using $O(\alpha\mathscr{M}(m))$ operations in $\mathbb{F}$. Then, in time $\mathscr{C}_{\text{inv}}(\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}, \alpha)$ we either assert that A' is not invertible or deduce a $\nabla_{\mathbb{M}^t_{\mathbf{Q}},\mathbb{M}_{\mathbf{P}}}$-generator for A'^{-1}, say $(\mathsf{G}'_{\text{inv}}, \mathsf{H}'_{\text{inv}})$. Finally, if A' is invertible then, using $\mathbb{M}^t_{\mathbf{P}} = \mathbb{Y}_{\mathbf{P}}^{-1}\,\mathbb{M}_{\mathbf{P}}\,\mathbb{Y}_{\mathbf{P}}$ and $\mathbb{M}_{\mathbf{Q}} = \mathbb{Y}_{\mathbf{Q}}\,\mathbb{M}^t_{\mathbf{Q}}\,\mathbb{Y}_{\mathbf{Q}}^{-1}$, we obtain a $\nabla_{\mathbb{M}_{\mathbf{Q}},\mathbb{M}^t_{\mathbf{P}}}$-generator $(\mathbb{Y}_{\mathbf{Q}}\mathsf{G}'_{\text{inv}}, \mathbb{Y}_{\mathbf{P}}\mathsf{H}'_{\text{inv}})$ for A^{-1} in time $O(\alpha\mathscr{M}(m))$. This proves the last claim. ∎

4.2. Reduction to the Hankel case. Our second reduction is less straightforward: we use Pan's idea of multiplicative transformation of operators [30] to reduce *basic* operators to an operator of Hankel type. There is some flexibility in the choice of the target operator, here the Sylvester operator $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}$; the only (natural) requirement is that this target operator remains invertible.

The following proposition summarizes the transformation process. Although the formulas are long, they describe simple processes: for an operation such as multiplication, inversion or system solving, this amounts to e.g. the analogue operation for an operator of Hankel type, several products with simple matrices derived from $\mathbf{P}$ and $\mathbf{Q}$, and $O(1)$ matrix-vector products with the input matrix or its transpose.

PROPOSITION 23. *Let $\mathcal{L} \in \{\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}, \Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}\}$, and suppose that $\mathscr{H}_{\mathbf{P}}$ and $\mathscr{H}_{\mathbf{Q}}$ hold. Then*

$$\mathscr{C}_{\text{mul}}(\mathcal{L}, \alpha, \beta) \leqslant \mathscr{C}_{\text{mul}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}, \alpha+2, \beta) + O\big(\mathscr{D}(\mathbf{P},\mathbf{Q}) + (\alpha+\beta)\mathscr{C}(\mathbf{P},\mathbf{Q})\big), \tag{5}$$

$$\mathscr{C}_{\text{solve}}(\mathcal{L}, \alpha) \leqslant \mathscr{C}_{\text{solve}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}, \alpha+2) + O\big(\mathscr{D}(\mathbf{P},\mathbf{Q}) + \alpha\mathscr{C}(\mathbf{P},\mathbf{Q})\big), \tag{6}$$

$$\mathscr{C}_{\text{inv}}(\mathcal{L}, \alpha) \leqslant \mathscr{C}_{\text{inv}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{m,1}}, \alpha+2) + O\big(\mathscr{D}(\mathbf{P},\mathbf{Q}) + \alpha\mathscr{C}(\mathbf{P},\mathbf{Q}) + \alpha^{\omega-1}m\big). \tag{7}$$

The proof of Proposition 23 will occupy the rest of this subsection. Before that, though, we point out that Theorem 21 follows directly from this proposition and Lemma 22. In particular, since $p \leqslant \mathscr{M}(p) \leqslant \mathscr{M}(m) + \mathscr{M}(n) \leqslant \mathscr{C}(\mathbf{P},\mathbf{Q})$ for $p = \max(m,n)$, overheads such as $O(\alpha\mathscr{M}(m))$ or $O(\alpha\mathscr{M}(p))$ that appear in that lemma are absorbed into terms such as $O(\alpha\mathscr{C}(\mathbf{P},\mathbf{Q}))$ that appear in the proposition.

Preliminaries. To establish each of the claims in the proposition above, it will be useful to have simple matrices L and R for pre-/post-multiplying A and whose displacement rank with respect to $\nabla_{\mathbb{Z}_{m,0},\mathbb{M}_{\mathbf{P}}}$ and $\nabla_{\mathbb{M}^t_{\mathbf{Q}},\mathbb{Z}^t_{n,1}}$, respectively, is small. Lemma 24 below shows that a possible choice, leading to displacement ranks at most 1, is

$$\mathsf{L} = \mathbb{J}_m\,\mathbb{W}^t_{\mathbf{P}}\,\mathbb{Y}^{-1}_{\mathbf{P}} \quad \text{and} \quad \mathsf{R} = \mathbb{Y}^{-1}_{\mathbf{Q}}\,\mathbb{W}_{\mathbf{Q}}\,\mathbb{J}_n.$$

In order to define generators for such matrices, we first rewrite the products $P = P_1 \cdots P_d$ and $Q = Q_1 \cdots Q_e$ as $P = p_0 + \cdots + p_{m-1}x^{m-1} + x^m$ and $Q = q_0 + \cdots + q_{n-1}x^{n-1} + x^n$, and let $\mathsf{m} = [p_0 \ \cdots \ p_{m-1}]^t$ and $\mathsf{n} = [q_0 \ \cdots \ q_{n-1}]^t$ be the coefficient

vectors of the polynomials $P - x^m$ and $Q - x^n$. Then, we let

$$\mathsf{t} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{F}^m, \quad \mathsf{u} = \mathbb{Y}_{\mathbf{P}}^{-1}\, \mathbb{W}_{\mathbf{P}}\, \mathsf{m} \in \mathbb{F}^m, \quad \mathsf{s} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{F}^n, \quad \mathsf{r} = -\mathbb{Y}_{\mathbf{Q}}^{-1}\, \mathbb{W}_{\mathbf{Q}}\, (\mathsf{n}+\mathsf{s}) \in \mathbb{F}^n.$$

LEMMA 24. *The relations* $\nabla_{\mathbb{Z}_{m,0},\mathbb{M}_{\mathbf{P}}}(\mathsf{L}) = \mathsf{t}\,\mathsf{u}^t$ *and* $\nabla_{\mathbb{M}_{\mathbf{Q}}^t,\mathbb{Z}_{n,1}^t}(\mathsf{R}) = \mathsf{r}\,\mathsf{s}^t$ *hold.*

Proof. According to the second part of Lemma 11, $\mathbb{Y}_{\mathbf{P}}^{-1}\,\mathbb{W}_{\mathbf{P}}$ is equal to the Krylov matrix $\mathbb{K}(\mathbb{M}_{\mathbf{P}}^t, \mathsf{w}, m)$, so that the columns of $\mathsf{L}^t = \mathbb{Y}_{\mathbf{P}}^{-1}\mathbb{W}_{\mathbf{P}}\,\mathbb{J}_m$ are $(\mathbb{M}_{\mathbf{P}}^t)^{m-1}\mathsf{w}, \ldots,$ $\mathbb{M}_{\mathbf{P}}^t\mathsf{w}, \mathsf{w}$. Hence only the first column of $\mathbb{M}_{\mathbf{P}}^t\,\mathsf{L}^t - \mathsf{L}^t\,\mathbb{Z}_{m,0}^t$ is nonzero, and it is equal to $\tilde{\mathsf{u}} := (\mathbb{M}_{\mathbf{P}}^t)^m\,\mathsf{w}$. After transposition, this leads to $\nabla_{\mathbb{Z}_{m,0},\mathbb{M}_{\mathbf{P}}}(\mathsf{L}) = -\mathsf{t}\,\tilde{\mathsf{u}}^t$. We can now check that $\tilde{\mathsf{u}} = -\mathsf{u}$ as follows. First, from Lemma 10 and since $\mathbb{Y}_{\mathbf{P}}$ is invertible, we see that $\tilde{\mathsf{u}} = \mathbb{Y}_{\mathbf{P}}^{-1}\,\mathbb{M}_{\mathbf{P}}^m\,\mathbb{Y}_{\mathbf{P}}\,\mathsf{w}$. Then, the shape of w implies that $\mathbb{Y}_{\mathbf{P}}\,\mathsf{w}$ is the vector v defined in the first part of Lemma 11, so that $\tilde{\mathsf{u}} = \mathbb{Y}_{\mathbf{P}}^{-1}\,\mathbb{M}_{\mathbf{P}}^m\,\mathsf{v}$. Now, the special shape of v implies that $\mathbb{M}_{\mathbf{P}}^m\,\mathsf{v}$ is the vector whose ith subvector of length m_i contains the coefficients of $x^m \bmod P_i = -(P - x^m) \bmod P_i$. In other words, $\mathbb{M}_{\mathbf{P}}^m\,\mathsf{v} = -\mathbb{W}_{\mathbf{P}}\,\mathsf{m}$. This shows that $\tilde{\mathsf{u}} = -\mathbb{Y}_{\mathbf{P}}^{-1}\,\mathbb{W}_{\mathbf{P}}\,\mathsf{m} = -\mathsf{u}$, which concludes the proof of the first relation.

For the second relation, the proof is the same, taking into account that we are now considering the operator $\nabla_{\mathbb{M}_{\mathbf{Q}}^t,\mathbb{Z}_{n,1}^t}$ and that $\mathbb{Z}_{n,1}^t = \mathbb{Z}_{n,0}^t + \mathbb{J}_n\,\mathsf{s}\mathsf{s}^t$. ∎

Proof of (5) *for* $\mathcal{L} = \nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}$. Let A be in $\mathbb{F}^{m\times n}$ and let (G,H) be a $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}$-generator for A. With $\mathsf{L}, \mathsf{R}, \mathsf{r}, \mathsf{s}, \mathsf{t}, \mathsf{u}$ as in the previous paragraph, define further

$$\mathsf{G}' = \left[\mathsf{t} \mid \mathsf{L}\,\mathsf{G} \mid \mathsf{L}\,\mathsf{A}\,\mathsf{r}\right] \quad \text{and} \quad \mathsf{H}' = \left[\mathsf{R}^t\,\mathsf{A}^t\,\mathsf{u} \mid \mathsf{R}^t\,\mathsf{H} \mid \mathsf{s}\right].$$

LEMMA 25. *The matrix* $\mathsf{A}' = \mathsf{L}\,\mathsf{A}\,\mathsf{R}$ *satisfies* $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t}(\mathsf{A}') = \mathsf{G}'\mathsf{H}'^t$.

Proof. Applying the general formula

$$\nabla_{\mathsf{M},\mathsf{N}}(\mathsf{ABC}) = \nabla_{\mathsf{M},\mathsf{Q}}(\mathsf{A})\mathsf{BC} + \mathsf{A}\nabla_{\mathsf{Q},\mathsf{R}}(\mathsf{B})\mathsf{C} + \mathsf{AB}\nabla_{\mathsf{R},\mathsf{N}}(\mathsf{C}) \tag{8}$$

(which follows directly from [33, Theorem 1.5.4]) and then using Lemma 24, we obtain

$$\begin{aligned}\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t}(\mathsf{L}\,\mathsf{A}\,\mathsf{R}) &= \nabla_{\mathbb{Z}_{m,0},\mathbb{M}_{\mathbf{P}}}(\mathsf{L})\,\mathsf{A}\,\mathsf{R} + \mathsf{L}\,\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}(\mathsf{A})\,\mathsf{R} + \mathsf{L}\,\mathsf{A}\,\nabla_{\mathbb{M}_{\mathbf{Q}}^t,\mathbb{Z}_{n,1}^t}(\mathsf{R}) \\ &= \mathsf{t}\,\mathsf{u}^t\,\mathsf{A}\,\mathsf{R} + \mathsf{L}\,\mathsf{G}\,\mathsf{H}^t\,\mathsf{R} + \mathsf{L}\,\mathsf{A}\,\mathsf{r}\,\mathsf{s}^t,\end{aligned}$$

which is the announced equality. ∎

To compute a product of the form AB with $\mathsf{B} \in \mathbb{F}^{n\times\beta}$, we first compute the matrices G' and H' described above. To obtain G', it suffices to set up the vectors r and Ar and to compute $\alpha + 1$ matrix-vector products by L. Given (G,H) and r, Corollary 14 implies that Ar is obtained in time

$$O(\mathscr{D}(\mathbf{P},\mathbf{Q}) + \alpha\mathscr{C}(\mathbf{P},\mathbf{Q})).$$

On the other hand, it follows from Lemmas 4 and 9 that the vector r is obtained in time $O(\mathscr{C}(\mathbf{Q}) + \mathscr{D}(\mathbf{Q}))$ and, after some precomputation of time $O(\mathscr{C}(\mathbf{P}) + \mathscr{D}(\mathbf{P}))$, that the $\alpha + 1$ products by L are obtained in time $O(\alpha\mathscr{C}(\mathbf{P}))$. Thus, overall, G' is obtained in time $O(\mathscr{D}(\mathbf{P},\mathbf{Q}) + \alpha\mathscr{C}(\mathbf{P},\mathbf{Q}))$.

We proceed similarly for H'. The only differences are that we have to do matrix-vector products involving R^t and A^t instead of L and A. For the former, Lemmas 4 and 9 show that after a precomputation of cost $O(\mathscr{C}(\mathbf{Q})+\mathscr{D}(\mathbf{Q}))$, the cost of one such product is $O(\mathscr{C}(\mathbf{Q}))$. For the latter, recall that, as pointed out in the introduction, from the given $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}$-generator of A, we can deduce in negligible time a $\nabla_{\mathbb{M}_{\mathbf{Q}},\mathbb{M}_{\mathbf{P}}^t}$-generator of A^t of the same length α. This allows us to do matrix-vector products with A^t with the same asymptotic cost

$$O(\mathscr{D}(\mathbf{Q},\mathbf{P}) + \alpha\mathscr{C}(\mathbf{Q},\mathbf{P})) = O(\mathscr{D}(\mathbf{P},\mathbf{Q}) + \alpha\mathscr{C}(\mathbf{P},\mathbf{Q}))$$

as for A. Thus, the overall cost for computing H' is asymptotically the same as for G'.

Let us now bound the cost of deducing the product AB from $\mathsf{G}', \mathsf{H}', \mathsf{B}$. Note first that under the coprimality assumptions $\mathscr{H}_{\mathbf{P}}$ and $\mathscr{H}_{\mathbf{Q}}$ the matrices $\mathbb{W}_{\mathbf{P}}$ and $\mathbb{W}_{\mathbf{Q}}$ are invertible, and so are L and R. Consequently, $\mathsf{AB} = \mathsf{L}^{-1}\mathsf{A}'\mathsf{R}^{-1}\mathsf{B}$ and it suffices to bound the cost of each of the three products $\mathsf{B}' := \mathsf{R}^{-1}\mathsf{B}$, $\mathsf{B}'' := \mathsf{A}'\mathsf{B}'$, and $\mathsf{L}^{-1}\mathsf{B}''$. By reusing the same precomputation as for H' and applying again Lemmas 4 and 9, we obtain B' for an additional cost of $O(\beta\mathscr{C}(\mathbf{Q}))$, via β matrix-vector products by R^{-1}. Then, Lemma 25 says that A' is a Hankel-like matrix for which a $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t}$-generator of length $\alpha+2$ is $(\mathsf{G}', \mathsf{H}')$. Hence the cost for deducing B'' from $\mathsf{G}', \mathsf{H}', \mathsf{B}'$ is at most $\mathscr{C}_{\text{mul}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t}, \alpha+2, \beta)$. Finally, $\mathsf{L}^{-1}\mathsf{B}''$ is obtained for an additional cost of $O(\beta\mathscr{C}(\mathbf{P}))$, by reusing the same precomputation as for G' and by performing β matrix-vector products by L^{-1}.

To summarize, we have shown that $(\mathsf{G}', \mathsf{H}')$ can be obtained in time $O(\mathscr{D}(\mathbf{P},\mathbf{Q}) + \alpha\mathscr{C}(\mathbf{P},\mathbf{Q}))$ and that AB can be deduced from $\mathsf{G}', \mathsf{H}', \mathsf{B}$ in time $\mathscr{C}_{\text{mul}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t}, \alpha+2, \beta) + O(\beta\mathscr{C}(\mathbf{P},\mathbf{Q}))$. Adding these two costs thus proves the bound (5) in Proposition 23 for $\mathcal{L} = \nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}$.

Proof of (6) for $\mathcal{L} = \nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}$. For $\mathsf{A} \in \mathbb{F}^{m\times n}$ given by some $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}$-generator (G, H) of length α and given b in $\mathbb{F}^m$, we now want to find a (nontrivial) solution of $\mathsf{Ax} = \mathsf{b}$, or determine that no solution exists.

Define $\mathsf{b}' = \mathsf{Lb}$. Because L and R are invertible matrices, solving the system $\mathsf{A}'\mathsf{x}' = \mathsf{b}'$ is equivalent to solving $\mathsf{Ax} = \mathsf{b}$, with then $\mathsf{x} = \mathsf{Rx}'$. As in the previous paragraph, we can compute a generator $(\mathsf{G}', \mathsf{H}')$ of A' for the operator $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t}$ in time $O(\mathscr{D}(\mathbf{P},\mathbf{Q})+\alpha\mathscr{C}(\mathbf{P},\mathbf{Q}))$. The cost of computing b' from b (and x from x') fits into the same bound, and solving the new system $\mathsf{A}'\mathsf{x}' = \mathsf{b}'$ takes time $\mathscr{C}_{\text{solve}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t}, \alpha+2)$. Summing these costs, we prove the second item in the proposition.

Proof of (7) for $\mathcal{L} = \nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}$. Assume now that $m = n$ and that $\mathsf{A} \in \mathbb{F}^{m\times m}$ is given by a $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}_{\mathbf{Q}}^t}$-generator (G, H) of length α. As before, L and R are invertible by assumption, so that A is invertible if and only if $\mathsf{A}' = \mathsf{L}\,\mathsf{A}\,\mathsf{R}$ is invertible. By Lemma 25 this matrix A' has displacement rank at most $\alpha+2$ for $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{m,1}^t}$. Thus, as explained in the introduction, if A' is invertible then its inverse has displacement rank at most $\alpha+2$ for $\nabla_{\mathbb{Z}_{m,1}^t,\mathbb{Z}_{m,0}}$. The next lemma shows that if $(\mathsf{G}'_{\text{inv}}, \mathsf{H}'_{\text{inv}})$ is a $\nabla_{\mathbb{Z}_{m,1}^t,\mathbb{Z}_{m,0}}$-generator for the inverse of A', then a $\nabla_{\mathbb{M}_{\mathbf{Q}}^t,\mathbb{M}_{\mathbf{P}}}$-generator for the inverse of A is given by the matrices

$$\mathsf{G}_{\text{inv}} = \left[\, \mathsf{r} \mid \mathsf{R}\,\mathsf{G}'_{\text{inv}} \mid \mathsf{R}\,\mathsf{A}'^{-1}\,\mathsf{t} \,\right] \quad \text{and} \quad \mathsf{H}_{\text{inv}} = \left[\, \mathsf{L}^t\,\mathsf{A}'^{-t}\,\mathsf{s} \mid \mathsf{L}^t\,\mathsf{H}'_{\text{inv}} \mid \mathsf{u} \,\right].$$

LEMMA 26. *The matrix A^{-1} satisfies $\nabla_{\mathbb{M}_{\mathbf{Q}}^t,\mathbb{M}_{\mathbf{P}}}(\mathsf{A}^{-1}) = \mathsf{G}_{\text{inv}}\mathsf{H}_{\text{inv}}^t$.*

Proof. As for Lemma 25 the proof follows from [33, Theorem 1.5.4] and Lemma 24:

$$\begin{aligned}\nabla_{\mathbb{M}^t_{\mathbf{Q}},\mathbb{M}_{\mathbf{P}}}(\mathsf{R}\,\mathsf{A}'^{-1}\,\mathsf{L}) &= \nabla_{\mathbb{M}^t_{\mathbf{Q}},\mathbb{Z}^t_{m,1}}(\mathsf{R})\,\mathsf{A}'^{-1}\,\mathsf{L} + \mathsf{R}\,\nabla_{\mathbb{Z}^t_{m,1},\mathbb{Z}_{m,0}}(\mathsf{A}'^{-1})\,\mathsf{L}\\ &\qquad + \mathsf{R}\,\mathsf{A}'^{-1}\,\nabla_{\mathbb{Z}_{m,0},\mathbb{M}_{\mathbf{P}}}(\mathsf{L})\\ &= \mathsf{r}\mathsf{s}^t\,\mathsf{A}'^{-1}\,\mathsf{L} + \mathsf{R}\,\mathsf{G}'_{\mathrm{inv}}\,(\mathsf{H}'_{\mathrm{inv}})^t\,\mathsf{L} + \mathsf{R}\,\mathsf{A}'^{-1}\,\mathsf{t}\,\mathsf{u}^t.\end{aligned}$$

To conclude, we remark that $\mathsf{A}^{-1} = \mathsf{R}\,\mathsf{A}'^{-1}\,\mathsf{L}$. ∎

The bound in (7) can now be established as follows. We begin by computing $(\mathsf{G}', \mathsf{H}')$, which by Lemma 25 is a $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{m,1}}$-generator of length $\alpha+2$ of A'; as shown in the previous paragraph, this is done in time $O(\mathscr{D}(\mathbf{P},\mathbf{Q})+\alpha\mathscr{C}(\mathbf{P},\mathbf{Q}))$. Then, in time $\mathscr{C}_{\mathrm{inv}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{m,1}}, \alpha+2)$ we either conclude that A' is singular, or produce a $\nabla_{\mathbb{Z}^t_{m,1},\mathbb{Z}_{m,0}}$-generator $(\mathsf{G}'_{\mathrm{inv}}, \mathsf{H}'_{\mathrm{inv}})$ of length $\alpha+2$ for the inverse of A'. Finally, if A' is invertible we proceed in two steps: First, using the same amount of time as for G' and H', we set up the matrices $\mathsf{G}_{\mathrm{inv}}$ and $\mathsf{H}_{\mathrm{inv}}$ introduced before Lemma 26. Then we reduce each of these two matrices to arrive at a generator of length α; this generator compression step can be done in time $O(\alpha^{\omega-1}m)$, in view of [33, Remark 4.6.7]. Overall, these costs add up to the result reported in (7).

Proof of (5)– (7) for $\mathcal{L} = \Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}$. We only sketch the proof in these cases. This time, we rely on the (easily verified) general formula

$$\Delta_{\mathsf{M},\mathsf{N}}(\mathsf{ABC}) = -\nabla_{\mathsf{M},\mathsf{P}}(\mathsf{A})\mathsf{BQC} + \mathsf{A}\Delta_{\mathsf{P},\mathsf{Q}}(\mathsf{B})\mathsf{C} + \mathsf{MAB}\nabla_{\mathsf{Q},\mathsf{N}}(\mathsf{C}).$$

We use it to write

$$\Delta_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}(\mathsf{L}\,\mathsf{A}\,\mathsf{R}) = -\nabla_{\mathbb{Z}_{m,0},\mathbb{M}_{\mathbf{P}}}(\mathsf{L})\,\mathsf{A}\,\mathbb{M}^t_{\mathbf{Q}}\,\mathsf{R} + \mathsf{L}\,\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}(\mathsf{A})\,\mathsf{R} + \mathbb{Z}_{m,0}\,\mathsf{L}\,\mathsf{A}\nabla_{\mathbb{M}^t_{\mathbf{Q}},\mathbb{Z}^t_{n,1}}(\mathsf{R});$$

this allows us to reduce questions (multiplication, linear system solving, inversion) related to A to the same questions for $\mathsf{A}' = \mathsf{L}\,\mathsf{A}\,\mathsf{R}$, where A' is given through a $\Delta_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}$-generator of length $\alpha+2$. Then, the relations $\mathbb{Z}^t_{n,1}\mathbb{J}_n = \mathbb{J}_n\mathbb{Z}_{n,1}$ and $\mathbb{Z}^t_{n,1} = \mathbb{Z}^{-1}_{n,1}$ imply

$$\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}(\mathsf{A}'\,\mathbb{J}_n) = -\Delta_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}(\mathsf{A}')\,\mathbb{Z}_{n,1}\,\mathbb{J}_n;$$

this allows us to reduce our problems to computations with the matrix $\mathsf{A}'\,\mathbb{J}_n$ given by means of a $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}$-generator of length $\alpha+2$.

For inversion (where $m = n$), inverting $\mathsf{A}'\,\mathbb{J}_n$ leads to a $\nabla_{\mathbb{Z}^t_{m,1},\mathbb{Z}_{m,0}}$-generator of $(\mathsf{A}'\,\mathbb{J}_m)^{-1} = \mathbb{J}_m\mathsf{A}'^{-1}$. Then, using the relations on $\mathbb{Z}_{m,1}$ given above, we obtain

$$\Delta_{\mathbb{Z}^t_{m,1},\mathbb{Z}_{m,0}}(\mathsf{A}'^{-1}) = \mathbb{Z}^t_{m,1}\,\mathbb{J}_m\,\nabla_{\mathbb{Z}^t_{m,1},\mathbb{Z}_{m,0}}(\mathbb{J}_m\mathsf{A}'^{-1})$$

and thus a $\Delta_{\mathbb{Z}^t_{m,1},\mathbb{Z}_{m,0}}$-generator of A'^{-1}. The general formula above further leads to

$$\begin{aligned}\Delta_{\mathbb{M}^t_{\mathbf{Q}},\mathbb{M}_{\mathbf{P}}}(\mathsf{R}\,\mathsf{A}'^{-1}\,\mathsf{L}) &= -\nabla_{\mathbb{M}^t_{\mathbf{Q}},\mathbb{Z}^t_{m,1}}(\mathsf{R})\,\mathsf{A}'^{-1}\,\mathbb{Z}_{m,0}\,\mathsf{L} + \mathsf{R}\,\Delta_{\mathbb{Z}^t_{m,1},\mathbb{Z}_{m,0}}(\mathsf{A}'^{-1})\,\mathsf{L}\\ &\qquad + \mathbb{M}^t_{\mathbf{Q}}\,\mathsf{R}\,\mathsf{A}'^{-1}\nabla_{\mathbb{Z}_{m,0},\mathbb{M}_{\mathbf{P}}}(\mathsf{L});\end{aligned}$$

since $\mathsf{R}\,\mathsf{A}'^{-1}\,\mathsf{L} = \mathsf{A}^{-1}$, this gives a $\Delta_{\mathbb{M}^t_{\mathbf{Q}},\mathbb{M}_{\mathbf{P}}}$-generator of A^{-1} whose length $\alpha+2$ is finally reduced to α.

In all cases Lemma 24 can be reused, and the requested complexity bounds are then derived in the same way as for the three cases above, up to a minor difference: we also need to take into account the cost of multiplication by $\mathbb{M}_{\mathbf{Q}}$ or its transpose with a vector. However, due to the sparse nature of this matrix, this cost is easily seen to be $O(n)$, so it does not impact the asymptotic estimate.

5. Multiplication algorithms. In this section, we prove bounds on the cost of the product AB, where A is a structured matrix in $\mathbb{F}^{m\times n}$ and B is an arbitrary matrix in $\mathbb{F}^{n\times\beta}$; this will prove the claims in Theorems 1 and 2 regarding multiplication.

THEOREM 27. *For any invertible operator $\mathcal{L}$ associated with $(\mathbf{P},\mathbf{Q})$, we can take*

$$\mathscr{C}_{\mathrm{mul}}(\mathcal{L},\alpha,\beta) = O\left(\frac{\beta'}{\alpha'}\mathscr{M}'_{\mathrm{mat}}\left(\frac{p}{\alpha'},\alpha'\right) + \mathscr{D}(\mathbf{P},\mathbf{Q}) + (\alpha+\beta)\mathscr{C}(\mathbf{P},\mathbf{Q})\right),$$

with $p=\max(m,n)$, $\alpha'=\min(\alpha,\beta)$, and $\beta'=\max(\alpha,\beta)$.

Using Lemma 22, it will be sufficient to consider multiplication for the basic operators associated with $(\mathbf{P},\mathbf{Q})$. Thus, throughout this section we let $\mathcal{L}:\mathbb{F}^{m\times n}\to\mathbb{F}^{m\times n}$ be one of the two operators $\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}$ and $\Delta_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}$ and, given $(\mathsf{G},\mathsf{H})\in\mathbb{F}^{m\times\alpha}\times\mathbb{F}^{n\times\alpha}$ and $\mathsf{B}\in\mathbb{F}^{n\times\beta}$, we show how to compute the product $\mathsf{AB}\in\mathbb{F}^{m\times\beta}$ with A the unique matrix in $\mathbb{F}^{m\times n}$ such that $\mathcal{L}(\mathsf{A})=\mathsf{GH}^t$.

In Subsection 5.1 we deal with the cases covered by Theorem 13; a key step of the proof (corresponding to the case $Q=x^n$) is deferred to Subsection 5.2 due to its length. We then extend our results to all remaining cases in Subsection 5.3.

In view of Theorem 21, it would actually be sufficient to assume that $\mathcal{L}$ is the Hankel operator $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}$. This would allow us to bypass most of the derivation in Subsection 5.3; however, such a restriction does not seem to lead to a better cost estimate than our direct approach, which we include for completeness.

5.1. Applying Theorem 13. Let us first assume that $\mathcal{L}=\nabla_{\mathbb{M}_{\mathbf{P}},\mathbb{M}^t_{\mathbf{Q}}}$. Given $\mathbf{P}$, $\mathbf{Q}$ and the generator (G,H), and a matrix $\mathsf{B}\in\mathbb{F}^{n\times\beta}$, Theorem 13 shows that AB can be computed as follows:

1. compute the polynomials P, Q, γ_k, η_k, Q^{-1} mod P_i for $k\leqslant\alpha$ and $i\leqslant d$;
2. compute the $n\times\beta$ matrix $\mathsf{B}'=\mathbb{X}_{\mathbf{Q}}\,\mathbb{Y}_{\mathbf{Q}}\,\mathsf{B}$;
3. compute the $m\times\beta$ matrix C given by
$$\mathsf{C}=\sum_{k\leqslant\alpha}\mathbb{M}_{\gamma_k,P,n}\,\mathbb{M}_{\eta_k,Q}\,\mathsf{B}';$$
4. return the $m\times\beta$ matrix $\mathbb{V}_{\mathbf{P},\mathbf{Q}}\,\mathbb{W}_{\mathbf{P}}\,\mathsf{C}$.

Proceeding as in the proof of Corollary 14, we see that Steps 1, 2, 4 can be completed in time $O(\mathscr{D}(\mathbf{P},\mathbf{Q})+(\alpha+\beta)\mathscr{C}(\mathbf{P},\mathbf{Q}))$. We are thus left with analyzing Step 3.

Let $\mathsf{b}_1,\dots,\mathsf{b}_\beta$ be the columns of B', and for $i=1,\dots,\beta$ let $B_i=\mathsf{pol}(\mathsf{b}_i)\in\mathbb{F}[x]_n$ be the polynomial whose coefficients are the entries of b_i. In polynomial terms, given $B_1,\dots,B_\beta$ in $\mathbb{F}[x]_n$, we want to compute $C_1,\dots,C_\beta$ in $\mathbb{F}[x]_m$ such that

$$C_i=\sum_{k\leqslant\alpha}\gamma_k(\eta_k B_i \bmod Q) \bmod P$$

for $i\leqslant\beta$; then, the m coefficients of C_i will make up the ith column of the matrix C.

To compute C_i, we can compute the sum first, and then reduce it modulo P. The core problem is thus to compute

$$R_i=\sum_{k\leqslant\alpha}\gamma_k(\eta_k B_i \bmod Q)$$

in $\mathbb{F}[x]_{m+n-1}$ for all $i\leqslant\beta$; the cost is given in Theorem 31 of Subsection 5.3 below: writing $\alpha'=\min(\alpha,\beta)$ and $\beta'=\max(\alpha,\beta)$, one can compute $R_1,\dots,R_\beta$ in time

$$O\left(\frac{\beta'}{\alpha'}\mathscr{M}'_{\mathrm{mat}}\left(\frac{p}{\alpha'},\alpha'\right)\right). \tag{9}$$

Finally, since R_i has degree less than $m+n-1$ and since P has degree m, computing each $C_i = R_i \bmod P$ takes time $O(\mathscr{M}(m)+\mathscr{M}(n)) \subset O(\mathscr{M}(p))$, so we can deduce $C_1, \ldots, C_\beta$ from $R_1, \ldots, R_\beta$ for a negligible total cost of $O(\alpha\mathscr{M}(p))$.

In the case $\mathcal{L} = \Delta_{\mathbb{M}_\mathbf{P}, \mathbb{M}^t_\mathbf{Q}}$, the approach is almost the same. A minor difference is that we replace $\mathbb{V}_{\mathbf{P},\mathbf{Q}}$ by $\mathbb{V}'_{\mathbf{P},\mathbf{Q}}$: this has no influence on the cost. The other difference is the matrix $\mathbb{J}_n$ appearing in the inversion formula for $\Delta_{\mathbb{M}_\mathbf{P}, \mathbb{M}^t_\mathbf{Q}}$ (second part of Theorem 13); in polynomial terms, this now leads us to compute the sums

$$R'_i = \sum_{k \leqslant \alpha} \gamma_k \, \mathsf{rev}(\eta_k B_i \bmod Q, n-1)$$

for $i = 1, \ldots, \beta$, and then to reduce each of them modulo P.

This problem is actually an instance of the question treated above, for we have

$$\mathsf{rev}(R'_i, m+n-2) = \sum_{k \leqslant \alpha} \mathsf{rev}(\gamma_k, m-1)(\eta_k B_i \bmod Q).$$

Since knowing $\mathsf{rev}(R'_i, m+n-2)$ gives us R'_i for free, we can appeal here as well to Theorem 31 to compute R'_i, using $\mathsf{rev}(\gamma_k, m-1)$ instead of γ_k. The cost estimate is the same as before, namely, (9), thus leading to Theorem 27 regarding a product of the form AB for the operator $\mathcal{L} = \Delta_{\mathbb{M}_\mathbf{P}, \mathbb{M}^t_\mathbf{Q}}$.

5.2. Computing the R_i's when $Q = x^n$. The previous subsection has shown that for both Sylvester's and Stein's displacement operators, computing the product AB essentially reduces to the following basic problem: given as input

- a monic polynomial Q of degree n,
- polynomials $U_1, \ldots, U_\alpha$ in $\mathbb{F}[x]_m$ and $V_1, \ldots, V_\alpha, W_1, \ldots, W_\beta$ in $\mathbb{F}[x]_n$,

compute

$$R_i = \sum_{k \leqslant \alpha} U_k(V_k W_i \bmod Q) \qquad \text{for } i = 1, \ldots, \beta.$$

The naive algorithm computes all sums independently; writing $p = \max(m,n)$, its cost is $O(\alpha\beta\mathscr{M}(p))$. In this subsection we establish the following improved complexity estimate in the special case where $Q = x^n$. (This result will be further extended to the case of a general Q of degree n in Subsection 5.3.)

THEOREM 28. *Assume $Q = x^n$ and $\alpha \leqslant n$, and let $p = \max(m,n)$, $\alpha' = \min(\alpha,\beta)$, and $\beta' = \max(\alpha,\beta)$. Then one can compute the polynomials $R_1, \ldots, R_\beta$ defined above in time*

$$O\left(\frac{\beta'}{\alpha'}\mathscr{M}'_{\mathrm{mat}}\left(\frac{p}{\alpha'}, \alpha'\right)\right).$$

To prove Theorem 28, we first rephrase our problem in polynomial matrix terms. Let

$$\mathsf{U} \in \mathbb{F}[x]_m^{\alpha\times 1}, \quad \mathsf{V} \in \mathbb{F}[x]_n^{\alpha\times 1}, \quad \mathsf{W} \in \mathbb{F}[x]_n^{\beta\times 1}$$

be the polynomial vectors with respective entries (U_i), (V_i) and (W_i). Then, our problem amounts to computing the polynomial row vector $\mathsf{R} \in \mathbb{F}[x]_{m+n-1}^{1\times\beta}$ such that

$$\mathsf{R} = \mathsf{U}^t(\mathsf{V}\mathsf{W}^t \bmod x^n).$$

We first solve that problem in the special case where $\alpha = \beta$ (this is where most of the difficulty takes place); then, we reduce the other cases of arbitrary α, β to this case.

Case $\alpha = \beta$. The following lemma is the basis of our algorithm in this case; it is taken from the proof of [4, Lemma 12].

LEMMA 29. *Let α, γ, ν be in $\mathbb{N}_{>0}$ with ν even. Let V, W be in $\mathbb{F}[x]_\nu^{\alpha\times\gamma}$ and define*

$$\mathsf{V}_0 = \mathsf{V} \bmod x^{\nu/2}, \quad \mathsf{V}_1 = \mathsf{V} \operatorname{div} x^{\nu/2},$$

$$\mathsf{W}_0 = \mathsf{W} \bmod x^{\nu/2}, \quad \mathsf{W}_1 = \mathsf{W} \operatorname{div} x^{\nu/2}.$$

Then the matrices $[\mathsf{V}_0\ \mathsf{V}_1]$ and $[\mathsf{W}_1\ \mathsf{W}_0]$ are in $\mathbb{F}[x]_{\nu/2}^{\alpha\times 2\gamma}$, and we have

$$\mathsf{V}\mathsf{W}^t \bmod x^\nu = \mathsf{V}_0\mathsf{W}_0^t + x^{\nu/2}\big([\mathsf{V}_0\ \mathsf{V}_1][\mathsf{W}_1\ \mathsf{W}_0]^t \bmod x^{\nu/2}\big).$$

We will want to apply this lemma recursively, starting from $\nu = n$ and $\gamma = 1$. It will be convenient to have both n and α be powers of two.

- If n is not a power of two, we define $\overline{n} = 2^{\lceil \log n\rceil}$ and $\delta = \overline{n} - n$. One may then check that $a \bmod b = x^{-\delta}\big((x^\delta a) \bmod (x^\delta b)\big)$ for any a and nonzero b in $\mathbb{F}[x]$; applying this identity componentwise to the definition of R gives

$$\mathsf{R} = x^{-\delta}\,\mathsf{U}^t\big(\mathsf{V}(x^\delta\,\mathsf{W})^t \bmod x^{\overline{n}}\big), \tag{10}$$

  where V and $x^\delta\mathsf{W}$ have degree less than $\overline{n}$.
- If α is not a power of two, we define $\overline{\alpha} = 2^{\lceil \log \alpha\rceil}$ and $\mu = \overline{\alpha} - \alpha$. Then, we can introduce μ dummy polynomials (U_i), (V_i) and (W_i), all equal to zero, without affecting the value of $R_1, \ldots, R_\alpha$.

The resulting algorithm is given in Figure 1, with a top-level procedure mul that handles the cases when n or α are not powers of two by defining $\overline{n}$ and $\overline{\alpha}$ as above, and a main recursive procedure mul_rec. We initially set $\nu = \overline{n}$ and $\gamma = 1$; through each recursive call, the width γ of the matrices V and W is doubled, while their degree ν is divided by two, so that the invariant $\gamma\nu = \overline{n}$ is maintained.

LEMMA 30. *Let $p = \max(m, n)$. Algorithm* mul *of Figure 1 works correctly in time*

$$O\left(\mathscr{M}'_{\mathrm{mat}}\left(\frac{p}{\alpha}, \alpha\right)\right).$$

Proof. Correctness is a direct consequence of Lemma 29 and the discussion that followed it. For the cost analysis, we let $C(k)$ denote the cost of mul_rec called upon parameters $\overline{\alpha}$ and γ such that $\overline{\alpha} = 2^k\gamma$. Note that the total cost of mul will then be at most $C(\kappa)$, with $\kappa = \lceil \log \alpha\rceil$. In particular, below, we always have $k \leqslant \kappa$. Another useful remark is that we always have $\nu \leqslant 2p/\gamma$, since $\nu = \overline{n}/\gamma$ and $\overline{n} \leqslant 2n \leqslant 2p$.

If $k = 0$, we have $\gamma = \overline{\alpha}$, so we are in the base case of the algorithm, where V and W are in $\mathbb{F}[x]_\nu^{\overline{\alpha}\times\overline{\alpha}}$. We first compute $\mathsf{R}' = \mathsf{V}\mathsf{W}^t \bmod x^\nu$ in time $\mathscr{M}_{\mathrm{mat}}(\nu, \overline{\alpha})$. Then, given U in $\mathbb{F}[x]_m^{\overline{\alpha}\times 1}$ and R' in $\mathbb{F}[x]_\nu^{\overline{\alpha}\times\overline{\alpha}}$, we obtain $\mathsf{R} = \mathsf{U}^t\mathsf{R}'$ in two steps as follows: rewriting U^t in the form $\mathsf{U}^t = [1\ x^c\ x^{2c}\ \cdots\ x^{(\overline{\alpha}-1)c}]\mathsf{U}'$ with $c = \lfloor m/\overline{\alpha}\rfloor$ and $\mathsf{U}' \in \mathbb{F}[x]_c^{\overline{\alpha}\times\overline{\alpha}}$, we compute first $\mathsf{Q} = \mathsf{U}'\mathsf{R}'$ in time $\mathscr{M}_{\mathrm{mat}}(\max\{c, \nu\}, \overline{\alpha})$, and then deduce R from Q using at most $\overline{\alpha}^2(c + \nu - 1)$ additions in $\mathbb{F}$. In summary,

$$\begin{aligned} C(0) &\leqslant \mathscr{M}_{\mathrm{mat}}(\nu, \overline{\alpha}) + \mathscr{M}_{\mathrm{mat}}(\max\{c, \nu\}, \overline{\alpha}) + \overline{\alpha}^2(c + \nu - 1) \\ &\leqslant 2\mathscr{M}_{\mathrm{mat}}\left(\frac{2p}{\overline{\alpha}}, \overline{\alpha}\right) + 3\,\overline{\alpha}\,p, \end{aligned}$$

using $c \leqslant m/\overline{\alpha} \leqslant p/\overline{\alpha}$ and $\nu = \overline{n}/\overline{\alpha} \leqslant 2p/\overline{\alpha}$. Therefore, there is a constant c_0 with

$$C(0) \leqslant c_0 \cdot \mathscr{M}_{\mathrm{mat}}\Big(\frac{p}{\overline{\alpha}}, \overline{\alpha}\Big). \tag{11}$$

Algorithm $\mathsf{mul\_rec}(\mathsf{U}, \mathsf{V}, \mathsf{W}, m, \nu, \overline{\alpha}, \gamma)$

Input: $\mathsf{U} \in \mathbb{F}[x]_m^{\overline{\alpha}\times 1}$, $\mathsf{V}, \mathsf{W} \in \mathbb{F}[x]_\nu^{\overline{\alpha}\times\gamma}$
Assumptions: $\overline{\alpha}$, γ and ν are powers of two and $\gamma \leqslant \overline{\alpha}$
Output: $\mathsf{R} \in \mathbb{F}[x]_{m+\nu-1}^{1\times\overline{\alpha}}$ such that $\mathsf{R} = \mathsf{U}^t(\mathsf{V}\mathsf{W}^t \bmod x^\nu)$.

```
if γ = α̅ then
    R' := VW^t mod x^ν
    R := U^t R'
else
    ν' := ν/2; γ' = 2γ
    V_0 := V mod x^{ν'}; V_1 := V div x^{ν'}; V' := [V_0 V_1]
    W_0 := W mod x^{ν'}; W_1 := W div x^{ν'}; W' := [W_1 W_0]
    R' := mul_rec(U, V', W', m, ν', α̅, γ')
    R := U^t V_0 W_0^t + x^{ν'} R'
return R.
```

Algorithm $\mathsf{mul}(\mathsf{U}, \mathsf{V}, \mathsf{W}, m, n, \alpha)$

Input: $\mathsf{U} \in \mathbb{F}[x]_m^{\alpha\times 1}$, $\mathsf{V}, \mathsf{W} \in \mathbb{F}[x]_n^{\alpha\times 1}$
Assumption: $\alpha \leqslant n$
Output: $\mathsf{R} \in \mathbb{F}[x]_{m+n-1}^{1\times\alpha}$ such that $\mathsf{R} = \mathsf{U}^t(\mathsf{V}\mathsf{W}^t \bmod x^n)$.

```
n̅ := 2^⌈log(n)⌉; δ := n̅ − n
α̅ := 2^⌈log(α)⌉; μ := α̅ − α
U̅ := U augmented with μ zero rows
V̅ := V augmented with μ zero rows
W̅ := x^δ W augmented with μ zero rows
R̅ := mul_rec(U̅, V̅, W̅, m, n̅, α̅, 1)
R := x^{−δ} R̅
return the first α entries of R.
```

FIG. 1. *Algorithms* mul *and* mul_rec.

Let us now bound $C(k)$ when $k \geqslant 1$. Given $\mathsf{V}, \mathsf{W} \in \mathbb{F}[x]_\nu^{\overline{\alpha}\times\gamma}$, we can compute V_0, V_1, W_0, W_1 for free. Then R' is computed recursively in time $C(k-1)$, and it remains to bound the cost of producing the result as $\mathsf{R} = \mathsf{Q} + x^{\nu'}\mathsf{R}'$, with $\mathsf{Q} = \mathsf{U}^t\mathsf{V}_0\mathsf{W}_0^t$.

For now let us write $D(k)$ for the cost of computing Q. Given R' in $\mathbb{F}[x]_{m+\nu'-1}^{1\times\overline{\alpha}}$ with $\nu' = \nu/2$ and Q in $\mathbb{F}[x]_{m+\nu-1}^{1\times\overline{\alpha}}$, we can add them together using at most $\overline{\alpha}(m + \nu' - 2)$ additions in $\mathbb{F}$. Since $\nu' = \overline{n}/2\gamma \leqslant n/\gamma \leqslant n$, we deduce that for $k \geqslant 1$,

$$C(k) \leqslant C(k-1) + D(k) + 2\,\overline{\alpha}\,p. \tag{12}$$

In order to bound $D(k)$ let us rewrite as before U^t as $\mathsf{U}^t = [1\ x^c\ x^{2c}\ \cdots\ x^{(\gamma-1)c}]\mathsf{U}'$ with $c = \lfloor m/\gamma \rfloor$ and $\mathsf{U}' \in \mathbb{F}[x]_c^{\gamma\times\overline{\alpha}}$; we compute Q as $\mathsf{Q} = [1, x^c, x^{2c}, \ldots, x^{(\gamma-1)c}](\mathsf{U}'\mathsf{V}_0\mathsf{W}_0^t)$. The product $\mathsf{U}'\mathsf{V}_0\mathsf{W}_0^t$ involves three polynomial matrices of respective dimensions $\gamma\times\overline{\alpha}$, $\overline{\alpha}\times\gamma$, $\gamma\times\overline{\alpha}$, with entries of degree less than $\max\{c, \nu'\}$. Furthermore, since $c \leqslant m/\gamma$ and, as seen above, $\nu' \leqslant n/\gamma$, we have $\max\{c, \nu'\} \leqslant p/\gamma$. Since $\gamma \leqslant \overline{\alpha}$, we can proceed as in the second case considered in the proof of [4, Lemma 7] to show that $\mathsf{U}'\mathsf{V}_0\mathsf{W}_0^t$ can be evaluated as $(\mathsf{U}'\mathsf{V}_0)\mathsf{W}_0^t$ in time

$$\frac{\overline{\alpha}}{\gamma}\,\mathscr{M}_{\mathrm{mat}}\left(\frac{p}{\gamma}, \gamma\right) + 2\,\overline{\alpha}\,p + \frac{\overline{\alpha}}{\gamma}\,\mathscr{M}_{\mathrm{mat}}\left(\frac{2p}{\gamma}, \gamma\right).$$

Since $\mathsf{U}'\mathsf{V}_0\mathsf{W}_0^t$ has dimensions $\gamma \times \overline{\alpha}$ and degree less than $3p/\gamma$, reconstructing Q from that product requires at most $\gamma\overline{\alpha} \cdot 3p/\gamma = 3\,\overline{\alpha}\,p$ additions in $\mathbb{F}$. Thus, using $\overline{\alpha} = 2^k\gamma$,

$$D(k) \leqslant 2^k \left(\mathscr{M}_{\mathrm{mat}} \left(\frac{2^k p}{\overline{\alpha}}, \frac{\overline{\alpha}}{2^k} \right) + \mathscr{M}_{\mathrm{mat}} \left(\frac{2^{k+1} p}{\overline{\alpha}}, \frac{\overline{\alpha}}{2^k} \right) \right) + 5\,\overline{\alpha}\,p.$$

Combining this bound with (12) and the fact that $\mathscr{M}_{\mathrm{mat}}(2d, n) = O(\mathscr{M}_{\mathrm{mat}}(d, n))$, we deduce that there exists a constant c_1 such that for all $k \geqslant 1$,

$$C(k) \leqslant C(k-1) + c_1 \cdot 2^k \mathscr{M}_{\mathrm{mat}} \left(\frac{2^k p}{\overline{\alpha}}, \frac{\overline{\alpha}}{2^k} \right). \tag{13}$$

Taking all $k = 0, \ldots, \kappa$ into account, we deduce from (11) and (13) that the total time of mul is bounded as

$$C(\kappa) \leqslant \max\{c_0, c_1\} \cdot \sum_{k=0}^{\kappa} 2^k \mathscr{M}_{\mathrm{mat}} \left(\frac{2^k p}{\overline{\alpha}}, \frac{\overline{\alpha}}{2^k} \right) \in O\left(\mathscr{M}'_{\mathrm{mat}} \left(\frac{p}{\overline{\alpha}}, \overline{\alpha} \right) \right).$$

The conclusion follows from the facts that $p/\overline{\alpha} \leqslant p/\alpha$ and that $\overline{\alpha} < 2\alpha$. ∎

Case $\alpha < \beta$. In this situation, we split the vector W into vectors $\mathsf{W}_1, \ldots, \mathsf{W}_c$, with each W_i in $\mathbb{F}[x]_n^{\alpha \times 1}$ and $c = \lceil \beta/\alpha \rceil$ (so W_c may be padded with zeros). Then, algorithm mul is applied c times (namely, to the (U, V, W_i)) and by Lemma 30 we obtain $R_1, \ldots, R_\beta$ in time

$$O\left(\frac{\beta}{\alpha} \mathscr{M}'_{\mathrm{mat}} \left(\frac{p}{\alpha}, \alpha \right) \right).$$

Case $\beta < \alpha$. In this case, we split the vectors U and V into $\mathsf{U}_1, \ldots, \mathsf{U}_c$ and $\mathsf{V}_1, \ldots, \mathsf{V}_c$, with each U_i and V_i in $\mathbb{F}[x]_n^{\beta \times 1}$ and $c = \lceil \alpha/\beta \rceil$. As before, algorithm mul is applied c times, but now to the $(\mathsf{U}_i, \mathsf{V}_i, \mathsf{W})$, for a cost of

$$O\left(\frac{\alpha}{\beta} \mathscr{M}'_{\mathrm{mat}} \left(\frac{p}{\beta}, \beta \right) \right);$$

the c results thus produced are then added, for a negligible cost of $O(\alpha p)$.

In the next subsection, we denote by $\mathsf{mul}(\mathsf{U}, \mathsf{V}, \mathsf{W}, m, n, \alpha, \beta)$ the algorithm that handles all possible cases for $\alpha \leqslant n$, following the discussion in the above paragraphs.

5.3. Computing the R_i's in the general case. We now address the case of an arbitrary Q. Given such a Q of degree n in $\mathbb{F}[x]$, as well as $U_1, \ldots, U_\alpha$ in $\mathbb{F}[x]_m$, $V_1, \ldots, V_\alpha$ in $\mathbb{F}[x]_n$ and $W_1, \ldots, W_\beta$ in $\mathbb{F}[x]_n$, we recall that our goal is to compute

$$R_i = \sum_{k \leqslant \alpha} U_k (V_k W_i \bmod Q), \quad i = 1, \ldots, \beta.$$

THEOREM 31. *Assume $\alpha \leqslant n$, and let $p = \max(m, n)$, $\alpha' = \min(\alpha, \beta)$, and $\beta' = \max(\alpha, \beta)$. Then one can compute the polynomials $R_1, \ldots, R_\beta$ in time*

$$O\left(\frac{\beta'}{\alpha'} \mathscr{M}'_{\mathrm{mat}} \left(\frac{p}{\alpha'}, \alpha' \right) \right).$$

To prove Theorem 31, the idea is to use the well-known fact that Euclidean division reduces to power series inversion and multiplication; in this way, the proof of Theorem 31 will reduce to that of Theorem 28.

The first step is to replace remainders by quotients. By applying the Euclidean division equality $V_kW_i = (V_kW_i \text{ div } Q)Q + (V_kW_i \bmod Q)$ and defining

$$S_i = \sum_{k\leqslant\alpha} U_k(V_kW_i \text{ div } Q) \quad \text{for } i \leqslant \beta \quad \text{and} \quad T = \sum_{k\leqslant\alpha} U_kV_k,$$

we easily deduce the following lemma.

LEMMA 32. *$R_i = TW_i - QS_i$ for $i \leqslant \beta$.*

The main issue in our algorithm will be the computation of $S_1, \ldots, S_\beta$; we will actually compute their reverse polynomials. Observe that U_i has degree at most $m-1$, V_kW_i div Q has degree at most $n-2$, and S_i has degree at most $m+n-3$. We thus introduce the polynomials $\widetilde{S}_i = \mathsf{rev}(S_i, m+n-3)$; knowing those, we can recover the polynomials S_i for free. For $k \leqslant \alpha$ and $i \leqslant \beta$, let us also define

$$\widetilde{U_k} = \mathsf{rev}(U_k, m-1),\ \widetilde{V_k} = \frac{\mathsf{rev}(V_k, n-1)}{\mathsf{rev}(Q, n)} \bmod x^{n-1},\ \widetilde{W_i} = \mathsf{rev}(W_i, n-1) \bmod x^{n-1}.$$

All these polynomials are related by the following formula.

LEMMA 33. *$\widetilde{S}_i = \sum_{k\leqslant\alpha} \widetilde{U_k}(\widetilde{V_k}\widetilde{W_i} \bmod x^{n-1})$ for $i \leqslant \beta$.*

Proof. Using that $\mathsf{rev}(ab, r+s) = \mathsf{rev}(a, r)\mathsf{rev}(b, s)$ for any polynomials $a, b \in \mathbb{F}[x]$ of respective degrees r and s, we deduce from the definitions of S_i and $\widetilde{S}_i$ that

$$\widetilde{S}_i = \sum_{k\leqslant\alpha} \widetilde{U_k}\mathsf{rev}(V_kW_i \text{ div } Q, n-2).$$

Following [12, p. 258], we conclude with the equalities

$$\begin{aligned}\mathsf{rev}(V_kW_i \text{ div } Q, n-2) &= \frac{\mathsf{rev}(V_kW_i, 2n-2)}{\mathsf{rev}(Q, n)} \bmod x^{n-1}\\ &= \frac{\mathsf{rev}(V_k, n-1)\,\mathsf{rev}(W_i, n-1)}{\mathsf{rev}(Q, n)} \bmod x^{n-1}\\ &= \widetilde{V_k}\widetilde{W_i} \bmod x^{n-1}.\end{aligned}$$

∎

Figure 2 details the algorithm we just sketched. To prove Theorem 31, it suffices to observe that the cost boils down to one call to mul plus $O((\alpha+\beta)\mathscr{M}(p))$ for all other operations, and then to apply Theorem 28.

6. Algorithms for inversion and system solving. We conclude in this section by establishing the cost bounds announced in Theorem 1 for structured inversion and structured system solving. We rely on Theorem 21 from Section 4 to reduce our task to the case of the Hankel operator $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}$, then use the fast multiplication algorithm of Section 5 to speed up the Morf / Bitmead-Anderson (MBA) approach.

THEOREM 34. *For any invertible operator $\mathcal{L}$ associated with $(\mathbf{P}, \mathbf{Q})$, we can take*

$$\mathscr{C}_{\text{inv}}(\mathcal{L}, \alpha) = O\left(\mathscr{M}''_{\text{mat}}\left(\frac{m}{\alpha}, \alpha\right)\right)$$

and

$$\mathscr{C}_{\text{solve}}(\mathcal{L}, \alpha) = O\left(\mathscr{M}''_{\text{mat}}\left(\frac{p}{\alpha}, \alpha\right)\right), \qquad p = \max(m, n).$$

Algorithm $\mathsf{mulQ}(\mathsf{U},\mathsf{V},\mathsf{W},m,n,\alpha,\beta,Q)$

Input: $\mathsf{U}\in\mathbb{F}[x]_m^{\alpha\times 1}$, $\mathsf{V}\in\mathbb{F}[x]_n^{\alpha\times 1}$, $\mathsf{W}\in\mathbb{F}[x]_n^{\beta\times 1}$
Q monic of degree n in $\mathbb{F}[x]$
Assumption: $\alpha\leqslant n$
Output: $\mathsf{R}\in\mathbb{F}[x]_{m+n-1}^{1\times\beta}$ such that $\mathsf{R}=\mathsf{U}^t(\mathsf{V}\mathsf{W}^t \bmod Q)$.

$\widetilde{\mathsf{U}} := [\mathsf{rev}(U_k, m-1), k=1,\dots,\alpha]$
$\widetilde{\mathsf{V}} := [\mathsf{rev}(V_k, n-1)/\mathsf{rev}(Q,n) \bmod x^{n-1}, k=1,\dots,\alpha]$
$\widetilde{\mathsf{W}} := [\mathsf{rev}(W_i, n-1) \bmod x^{n-1}, i=1,\dots,\beta]$
$\widetilde{\mathsf{S}} := \mathsf{mul}(\widetilde{\mathsf{U}},\widetilde{\mathsf{V}},\widetilde{\mathsf{W}},m,n-1,\alpha,\beta)$
$T := \sum_{k=1,\dots,\alpha} U_k V_k$
$\mathsf{R} := [TW_i - Q\mathsf{rev}(\widetilde{S}_i, m+n-3), i=1,\dots,\beta]$
return R.

FIG. 2. *Algorithm* mulQ.

Recall that Theorem 21 in Section 4 proves that we can take

$$\mathscr{C}_{\mathrm{inv}}(\mathcal{L},\alpha)\leqslant \mathscr{C}_{\mathrm{inv}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{m,1}^t},\alpha+2)+O\big(\mathscr{D}(\mathbf{P},\mathbf{Q})+\alpha\mathscr{C}(\mathbf{P},\mathbf{Q})+\alpha^{\omega-1}m\big)$$

and

$$\mathscr{C}_{\mathrm{solve}}(\mathcal{L},\alpha)\leqslant \mathscr{C}_{\mathrm{solve}}(\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t},\alpha+2)+O\big(\mathscr{D}(\mathbf{P},\mathbf{Q})+\alpha\mathscr{C}(\mathbf{P},\mathbf{Q})\big),$$

so we are left with estimating the cost of inversion and system solving for $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t}$. For this, we shall use the MBA approach with preconditioning introduced in [20] together with the "compression-free" formulas from [16] for generating the matrix inverse. Such formulas being expressed with products of the form (structured matrix) $\times$ (unstructured matrix), this provides a way to exploit our results for fast structured matrix multiplication from Section 5. Specifically, starting from a $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t}$-generator (G,H) of length α of $\mathsf{A}\in\mathbb{F}^{m\times n}$, we proceed in three steps as follows.

Preconditioning. From (G,H) we begin by deducing a generator of the preconditioned matrix

$$\widetilde{\mathsf{A}}=\mathbb{U}(\mathsf{v}_1)\mathsf{A}\mathbb{U}(\mathsf{v}_2)^t,$$

where each $\mathbb{U}(\mathsf{v}_i)$ is a unit upper triangular Toeplitz matrix defined by some random vector v_i; as shown in [21], such a preconditioning ensures that, with high probability, $\widetilde{\mathsf{A}}$ has *generic rank profile*, that is, denoting by $\widetilde{\mathsf{A}}_k$ the leading principal submatrix of $\widetilde{\mathsf{A}}$ of dimensions $k\times k$,

$$\det\big(\widetilde{\mathsf{A}}_k\big)\neq 0 \quad \text{for } k=1,\dots,\mathrm{rank}(\widetilde{\mathsf{A}}).$$

In our case, preconditioning will be done in such a way that the generator of $\widetilde{\mathsf{A}}$ corresponds not to $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t}$ but to another Hankel operator, namely $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,0}^t}$ (algorithm precond, Figure 3). The latter operator being singular but still *partly regular* [33, §4.5], such a generator will in fact be a triple $(\widetilde{\mathsf{G}},\widetilde{\mathsf{H}},\widetilde{\mathsf{u}})$, with $(\widetilde{\mathsf{G}},\widetilde{\mathsf{H}})$ of length $O(\alpha)$ and $\widetilde{\mathsf{u}}$ carrying the last row of $\widetilde{\mathsf{A}}$. Also, we will guarantee that the matrices $\widetilde{\mathsf{G}}$ and $\widetilde{\mathsf{H}}$ have the following special shape: their first α columns are precisely $\mathbb{U}(\mathsf{v}_1)\mathsf{G}$ and $\mathbb{U}(\mathsf{v}_2)\mathsf{H}$. These two requirements will be key to exploit the "compression-free" inversion scheme of [16, §4.3] and then to move back from, say, $\widetilde{\mathsf{A}}^{-1}$ to A^{-1}.

Computation of the leading principal inverse. Assuming that $\widetilde{\mathsf{A}}$ has generic rank profile and given $(\widetilde{\mathsf{G}}, \widetilde{\mathsf{H}}, \widetilde{\mathsf{u}})$, we now consider computing a compact representation of the inverse of its largest nonsingular leading principal submatrix, that is, of $\widetilde{\mathsf{A}}_r^{-1}$ with

$$r = \operatorname{rank}(\widetilde{\mathsf{A}}) = \operatorname{rank}(\mathsf{A}).$$

Since $\widetilde{\mathsf{A}}_r$ is structured with respect to $\nabla_{\mathbb{Z}_{r,0},\mathbb{Z}_{r,0}^t}$, its inverse is structured with respect to $\nabla_{\mathbb{Z}_{r,0}^t,\mathbb{Z}_{r,0}}$ and, as recalled in (1),

$$\operatorname{rank}\left(\nabla_{\mathbb{Z}_{r,0}^t,\mathbb{Z}_{r,0}}(\widetilde{\mathsf{A}}_r^{-1})\right) = \operatorname{rank}\left(\nabla_{\mathbb{Z}_{r,0},\mathbb{Z}_{r,0}^t}(\widetilde{\mathsf{A}})\right) =: \rho.$$

Therefore, our second step (detailed in Section 6.2) will be to compute a $\nabla_{\mathbb{Z}_{r,0}^t,\mathbb{Z}_{r,0}}$-generator of $\widetilde{\mathsf{A}}_r^{-1}$ of length ρ.

In fact, following [33, §5], we give an algorithm (lp_inv, Figure (6)) that does *not* assume $\widetilde{\mathsf{A}}$ has generic rank profile but discovers whether this is the case or not. It calls first an auxiliary routine, called largest, which returns the size ℓ of the largest leading principal submatrix $\widetilde{\mathsf{A}}_\ell$ having generic rank profile together with a generator of the inverse $\widetilde{\mathsf{A}}_\ell^{-1}$. Then, from such a generator, one can check efficiently whether ℓ equals $\operatorname{rank}(\widetilde{\mathsf{A}})$, which is a condition equivalent to $\widetilde{\mathsf{A}}$ having generic rank profile. Thus, overall, algorithm lp_inv generates $\widetilde{\mathsf{A}}_r^{-1}$ if and only if $\widetilde{\mathsf{A}}$ has generic rank profile (and reports 'failure' otherwise, since this provides a way of certifying whether preconditioning A was successful or not).

Algorithm largest calls a core recursive routine largest_rec, which can be seen as a combination of Kaltofen's algorithm Leading Principal Inverse [20] and [16, algorithm GenInvHL], which thus relies on products of the form (structured matrix) × (unstructured matrix). Also, the generating matrices $\widetilde{\mathsf{Y}}_\ell$ and $\widetilde{\mathsf{Z}}_\ell$ produced by algorithm largest are specified to have the following shape:

$$\widetilde{\mathsf{Y}}_\ell = -\widetilde{\mathsf{A}}_\ell^{-1}\widetilde{\mathsf{G}}_\ell, \qquad \widetilde{\mathsf{Z}}_\ell = \widetilde{\mathsf{A}}_\ell^{-t}\widetilde{\mathsf{H}}_\ell,$$

where $\widetilde{\mathsf{G}}_\ell$ is made from the first ℓ rows of $\widetilde{\mathsf{G}}$, and similarly for $\widetilde{\mathsf{H}}_\ell$. When $\ell = r$, the same specification is inherited by the generator $(\widetilde{\mathsf{Y}}_r, \widetilde{\mathsf{Z}}_r, \widetilde{\mathsf{v}})$ produced by algorithm lp_inv, whose matrices satisfy

$$\widetilde{\mathsf{Y}}_r = -\widetilde{\mathsf{A}}_r^{-1}\widetilde{\mathsf{G}}_r, \qquad \widetilde{\mathsf{Z}}_r = \widetilde{\mathsf{A}}_r^{-t}\widetilde{\mathsf{H}}_r$$

where $\widetilde{\mathsf{G}}_r$ and $\widetilde{\mathsf{H}}_r$ are the first r rows of $\widetilde{\mathsf{G}}$ and $\widetilde{\mathsf{H}}$, and $\widetilde{\mathsf{v}}^t$ is the first row of $\widetilde{\mathsf{A}}_r^{-1}$.

Generating inverses and solving linear systems. Given the rank r of $\widetilde{\mathsf{A}}$ and the $\nabla_{\mathbb{Z}_{r,0}^t,\mathbb{Z}_{r,0}}$-generator $(\widetilde{\mathsf{Y}}_r, \widetilde{\mathsf{Z}}_r, \widetilde{\mathsf{v}})$ of A_r^{-1} as above, it is immediate to decide whether $\widetilde{\mathsf{A}}$ and A are invertible and, if so, to deduce the $\nabla_{\mathbb{Z}_{m,0}^t,\mathbb{Z}_{m,1}}$-generator of A^{-1} given by

$$\mathsf{Y} = -\mathsf{A}^{-1}\mathsf{G}, \qquad \mathsf{Z} = \mathsf{A}^{-t}\mathsf{H}.$$

This corresponds to algorithm inv in Figure 7. Now, given an additional vector $\mathsf{b} \in \mathbb{F}^m$, we reduce the study of $\mathsf{Ax} = \mathsf{b}$ to that of the equivalent linear system $\widetilde{\mathsf{A}}\widetilde{\mathsf{x}} = \widetilde{\mathsf{b}}$, where $\widetilde{\mathsf{b}} = \mathbb{U}(\mathsf{v}_1)\mathsf{b}$ and for which an algorithm can be derived directly from the one in [21, §4], with the additional guarantee that if $\mathsf{b} = 0$ and the column rank of A is not full, then a nonzero solution x is obtained. This corresponds to algorithm solve in

Figure 8. (Clearly, both inv and solve are Las Vegas algorithms—"always correct, probably fast", thanks to the specification of algorithm lp_inv.)

The next three sections provide detailed descriptions of the algorithms mentioned above, namely precond, largest_rec, largest, lp_inv, inv, solve, together with their correctness and complexity proofs, thereby establishing Theorem 34.

6.1. Preconditioning. We precondition our structured matrix A as shown in Figure 3 below. Here and hereafter, $\mathsf{e}_{n,i}$ denotes the ith unit vector in $\mathbb{K}^n$ and, for any given $m \times n$ matrix M such that $n \geqslant \alpha$, we write $\mathsf{M}^{\mapsto\alpha}$ for the $m \times \alpha$ matrix obtained by keeping only the first α columns of M.

Algorithm $\mathsf{precond}(\mathsf{G}, \mathsf{H}, \mathsf{v}_1, \mathsf{v}_2)$

Input: $(\mathsf{G}, \mathsf{H}) \in \mathbb{F}^{m\times\alpha} \times \mathbb{F}^{n\times\alpha}$ such that $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}(\mathsf{A}) = \mathsf{G}\mathsf{H}^t$, $\mathsf{v}_1 \in \mathbb{F}^m$ and $\mathsf{v}_2 \in \mathbb{F}^n$.

Output: $(\widetilde{\mathsf{G}}, \widetilde{\mathsf{H}}) \in \mathbb{F}^{m\times(\alpha+4)} \times \mathbb{F}^{n\times(\alpha+4)}$ and $\widetilde{\mathsf{u}} \in \mathbb{F}^n$ such that

- $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,0}}(\widetilde{\mathsf{A}}) = \widetilde{\mathsf{G}}\widetilde{\mathsf{H}}^t$ for $\widetilde{\mathsf{A}} = \mathbb{U}(\mathsf{v}_1)\,\mathsf{A}\,\mathbb{U}(\mathsf{v}_2)^t$;
- $\widetilde{\mathsf{u}}^t$ is the last row of $\widetilde{\mathsf{A}}$;
- $\widetilde{\mathsf{G}}^{\mapsto\alpha} = \mathbb{U}(\mathsf{v}_1)\mathsf{G}$ and $\widetilde{\mathsf{H}}^{\mapsto\alpha} = \mathbb{U}(\mathsf{v}_2)\mathsf{H}$.

$\mathsf{G}_1 := \left[\mathbb{Z}_{m,0}\mathbb{J}_m\mathsf{v}_1 \mid -\mathsf{e}_{m,1}\right]$; $\mathsf{H}_1 := \left[\mathsf{e}_{m,m} \mid \mathbb{Z}^t_{m,0}\mathsf{v}_1\right]$

$\mathsf{G}_2 := \left[\mathbb{Z}^t_{n,1}\mathsf{v}_2 \mid -\mathsf{e}_{n,n}\right]$; $\mathsf{H}_2 := \left[\mathsf{e}_{n,1} \mid \mathbb{Z}_{n,0}\mathbb{J}_n\mathsf{v}_2\right]$

$\widetilde{\mathsf{G}} := \left[\mathbb{U}(\mathsf{v}_1)\mathsf{G} \mid \mathsf{G}_1 \mid \mathbb{U}(\mathsf{v}_1)\mathsf{A}\mathsf{G}_2\right]$; $\widetilde{\mathsf{H}} := \left[\mathbb{U}(\mathsf{v}_2)\mathsf{H} \mid \mathbb{U}(\mathsf{v}_2)\mathsf{A}^t\mathsf{H}_1 \mid \mathsf{H}_2\right]$

$\widetilde{\mathsf{u}} := \mathbb{U}(\mathsf{v}_2)\mathsf{A}^t\mathbb{U}(\mathsf{v}_1)^t\mathsf{e}_{m,m}$

return $(\widetilde{\mathsf{G}}, \widetilde{\mathsf{H}}, \widetilde{\mathsf{u}})$.

FIG. 3. *Algorithm* precond.

LEMMA 35. *Algorithm* precond *works correctly in time* $O(\alpha\mathsf{M}(p))$. *Furthermore, if the vectors* $\mathsf{v}_1 \in \mathbb{F}^m$ *and* $\mathsf{v}_2 \in \mathbb{F}^n$ *have their first entry equal to* 1 *and their remaining* $m + n - 2$ *entries chosen uniformly at random from a finite subset* $S \subset \mathbb{F}$, *then the matrix* $\widetilde{\mathsf{A}} = \mathbb{U}(\mathsf{v}_1)\mathsf{A}\mathbb{U}(\mathsf{v}_2)^t$ *has generic rank profile with probability at least*

$$1 - r(r+1)/|S|,$$

where $r = \operatorname{rank}\left(\widetilde{\mathsf{A}}\right) = \operatorname{rank}(\mathsf{A})$ *and* $|S|$ *is the cardinality of* S.

Proof. We start by checking $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,0}}(\widetilde{\mathsf{A}}) = \widetilde{\mathsf{G}}\widetilde{\mathsf{H}}^t$, from which correctness follows. Writing $\mathcal{L}_1 = \nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{m,0}}(\mathbb{U}(\mathsf{v}_1))$ and $\mathcal{L}_2 = \nabla_{\mathbb{Z}^t_{n,1},\mathbb{Z}^t_{n,0}}(\mathbb{U}(\mathsf{v}_2)^t)$ and applying (8) gives

$$\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,0}}(\widetilde{\mathsf{A}}) = \mathcal{L}_1\mathsf{A}\mathbb{U}(\mathsf{v}_2)^t + \mathbb{U}(\mathsf{v}_1)\mathsf{G}\mathsf{H}^t\mathbb{U}(\mathsf{v}_2)^t + \mathbb{U}(\mathsf{v}_1)\mathsf{A}\mathcal{L}_2,$$

and it remains to check that $\mathcal{L}_1 = \mathsf{G}_1\mathsf{H}_1^t$ and $\mathcal{L}_2 = \mathsf{G}_2\mathsf{H}_2^t$. Since $\mathbb{U}(\mathsf{v}_1)$ is upper triangular Toeplitz, the matrix $\mathcal{L}_1 = \mathbb{Z}_{m,0}\mathbb{U}(\mathsf{v}_1) - \mathbb{U}(\mathsf{v}_1)\mathbb{Z}_{m,0}$ is zero everywhere except on its last column, which is equal to $\mathbb{Z}_{m,0}\mathbb{J}_m\mathsf{v}_1$, and on its first row, which is equal to $-\mathsf{v}_1^t\mathbb{Z}_{m,0}$. Hence $\mathcal{L}_1 = \mathbb{Z}_{m,0}\mathbb{J}_m\mathsf{v}_1\,\mathsf{e}_{m,m}^t - \mathsf{e}_{m,1}\,(\mathbb{Z}^t_{m,0}\mathsf{v}_1)^t = \mathsf{G}_1\mathsf{H}_1^t$. For $\mathcal{L}_2$, we proceed similarly, by noting that $\mathcal{L}_2 = \mathbb{Z}^t_{n,1}\mathbb{U}(\mathsf{v}_2)^t - \mathbb{U}(\mathsf{v}_2)^t\mathbb{Z}^t_{n,0}$ is zero everywhere but on its first column and last row, equal to $\mathbb{Z}^t_{n,1}\mathsf{v}_2$ and $-\mathbb{Z}_{n,0}\mathbb{J}_n\mathsf{v}_2$, respectively.

Let us now bound the cost of deducing $\widetilde{\mathsf{G}}$, $\widetilde{\mathsf{H}}$, $\widetilde{\mathsf{u}}$ from G, H, v_1, v_2. First, we set up the $m \times 2$ matrices G_1 and H_1 and the $n \times 2$ matrices G_2 and H_2 in time $O(p)$. Then, since A satisfies $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,1}}(\mathsf{A}) = \mathsf{G}\mathsf{H}^t$, multiplying A or A^t be a single vector can be done in time $O(\alpha\mathsf{M}(p))$, using for example the reconstruction formula in [16, (21b)]. Hence we obtain the products $\mathsf{A}\mathsf{G}_2$, $\mathsf{A}^t\mathsf{H}_1$, and $\mathsf{A}^t\mathbb{U}(\mathsf{v}_1)^t\mathsf{e}_{m,m}$ in time $O(\alpha\mathsf{M}(p))$. Finally, since G and H have α columns each, it remains to multiply $O(\alpha)$ vectors by the triangular Toeplitz matrices $\mathbb{U}(\mathsf{v}_1)$ and $\mathbb{U}(\mathsf{v}_2)$, and this can be done in time $O(\alpha\mathsf{M}(p))$. Overall, the cost of the algorithm is thus bounded by $O(\alpha\mathsf{M}(p))$.

The probability analysis is due to Kaltofen and Saunders [21, Theorem 2]. ∎

6.2. Computation of the leading principal inverse. In order to generate the inverse of the largest nonsingular leading principal submatrix of A, we start with the generation of the largest leading principal submatrix having generic rank profile. This is done by algorithms largest_rec and largest displayed in Figures 4 and 5. In algorithm largest_rec the matrices A_{ij}, G_i, H_j for $1 \leqslant i, j \leqslant 2$ have dimensions $m_i \times m_j$, $m_i \times \alpha$, $m_j \times \alpha$, respectively, and correspond to the block partitions

$$\mathsf{A} = \begin{bmatrix} \mathsf{A}_{11} & \mathsf{A}_{12} \\ \mathsf{A}_{21} & \mathsf{A}_{22} \end{bmatrix}, \quad \mathsf{G} = \begin{bmatrix} \mathsf{G}_1 \\ \mathsf{G}_2 \end{bmatrix}, \quad \mathsf{H} = \begin{bmatrix} \mathsf{H}_1 \\ \mathsf{H}_2 \end{bmatrix}.$$

Similarly, u_{ij}^t denotes the last row of A_{ij}.

LEMMA 36. *Algorithm* largest_rec *is correct and if $m = n$ is an integer power of two, then its cost is*

$$O\Big(\mathscr{M}''_{\mathrm{mat}}\Big(\frac{m}{\alpha}, \alpha\Big)\Big).$$

Proof. Correctness follows directly from combining the analysis in [20, pp. 801–803] with the formulas for G_S, H_S, u_S, Y, Z, w given in [16, p. 287].

Assuming $m = n$ is a power of two, let $C(m, \alpha)$ denote the cost of algorithm largest_rec. We begin by showing that we can take

$$C(m, \alpha) = O(\alpha^\omega) \qquad \text{if } \alpha \leqslant m < 2\alpha.$$

Given G and H we compute the product $\mathsf{G}\mathsf{H}^t$ in time $O(\alpha^\omega)$. Then, starting from the last row of A and using the fact that for $i, j > 1$ the (i, j) entry of $\mathsf{G}\mathsf{H}^t$ equals $a_{i-1,j} - a_{i,j-1}$, we deduce all the entries of A in time $O(\alpha^2)$.

Let us now bound the cost when $m \geqslant 2\alpha$. Proceeding as in the proof of [16, Lemma 5], it is easily seen that one can compute some generators of length at most $\alpha+2$ for the matrices A_{21}, A_{12}^t, $\mathsf{A}_{11}^{-1}\mathsf{A}_{12}$, $\mathsf{A}_{22}^{-t}\mathsf{A}_{21}^t$ as well as the vectors u_{11}, u_{21}, u_{22}, u_S for a total time in $O(\alpha\,\mathsf{M}(m))$. The number of additions is in $O(\alpha\, m)$ and we also have to perform the four matrix products $\mathsf{A}_{21}\mathsf{Y}_{11}$, $\mathsf{A}_{12}^t\,\mathsf{Z}_{11}$, $(\mathsf{A}_{11}^{-1}\mathsf{A}_{12})\mathsf{Y}_\mathsf{S}$, and $(\mathsf{A}_{11}^{-T}\mathsf{A}_{21}^T)\mathsf{Z}_\mathsf{S}$. For example, A_{12} has displacement rank at most $\alpha + 2$ with respect to the operator $\nabla_{\mathbb{Z}_{m/2,0},\mathbb{Z}^t_{m/2,1}}$, so that using a decomposition of the form $\mathsf{A}_{12} = \mathsf{A}'_{12} + \mathsf{A}''_{12}$ with A'_{12} and A''_{12} of displacement ranks at most α and 2, respectively, we can evaluate the product $\mathsf{A}_{21}\mathsf{Y}_{11}$ in time $\mathscr{C}_{\mathrm{mul}}(\nabla_{\mathbb{Z}_{m/2,0},\mathbb{Z}^t_{m/2,1}}, \alpha, \alpha) + O(\alpha\,\mathsf{M}(m))$; by Theorem 2, this is in $O\big(\frac{m}{2\alpha}\mathscr{M}'_{\mathrm{mat}}(\frac{m}{2\alpha}, \alpha)\big)$. Adding up all these costs thus leads to

$$C(m, \alpha) = 2C\Big(\frac{m}{2}, \alpha\Big) + O\Big(\mathscr{M}'_{\mathrm{mat}}\Big(\frac{m}{2\alpha}, \alpha\Big)\Big) \qquad \text{if } m \geqslant 2\alpha.$$

Hence, for some constant c_0,

$$C(m, \alpha) \leqslant c_0 \left(\sum_{i=0}^{i_0-1} 2^i \mathscr{M}'_{\mathrm{mat}}\Big(\frac{m}{2^{i+1}\alpha}, \alpha\Big) + 2^{i_0}\alpha^\omega \right)$$

Algorithm $\mathsf{largest\_rec}(\mathsf{G}, \mathsf{H}, \mathsf{u})$

Input: $(\mathsf{G}, \mathsf{H}, \mathsf{u}) \in \mathbb{F}^{m\times\alpha} \times \mathbb{F}^{n\times\alpha} \times \mathbb{F}^{n}$ such that $\alpha \leqslant \min(m,n)$ and $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,0}}(\mathsf{A}) = \mathsf{G}\mathsf{H}^t$ and $\mathsf{u}^t = \mathsf{e}^t_{m,m}\mathsf{A}$ (the last row of A).

Output: $(\ell, \mathsf{Y}, \mathsf{Z}, \mathsf{v}) \in \mathbb{N} \times \mathbb{F}^{\ell\times\alpha} \times \mathbb{F}^{\ell\times\alpha} \times \mathbb{F}^{\ell}$ such that ℓ is the order of the largest leading principal submatrix A_ℓ of A having generic rank profile, $\mathsf{Y} = -\mathsf{A}_\ell^{-1}\mathsf{G}_\ell$, $\mathsf{Z} = \mathsf{A}_\ell^{-t}\mathsf{H}_\ell$ and $\mathsf{v} = \mathsf{A}_\ell^{-t}\mathsf{e}_{\ell,1}$ (the first row of A_ℓ^{-1}).

if $\min(m,n) < 2\alpha$ then
  compute A explicitly and then deduce ℓ, $\mathsf{Y}, \mathsf{Z}, \mathsf{v}$.
else
  $m_1 := \lceil m/2 \rceil$; $m_2 := \lfloor m/2 \rfloor$; $n_1 := \lceil n/2 \rceil$; $n_2 := \lfloor n/2 \rfloor$
  $(\ell_{11}, \mathsf{Y}_{11}, \mathsf{Z}_{11}, \mathsf{v}_{11}) := \mathsf{largest\_rec}(\mathsf{G}_1, \mathsf{H}_1, \mathsf{u}_{11})$
  if $\ell_{11} < \min(m_1, n_1)$ then
    $(\ell, \mathsf{Y}, \mathsf{Z}, \mathsf{v}) := (\ell_{11}, \mathsf{Y}_{11}, \mathsf{Z}_{11}, \mathsf{v}_{11})$
  else
    $\mathsf{G}_\mathsf{S} := \mathsf{G}_2 + \mathsf{A}_{21}\mathsf{Y}_{11}$; $\mathsf{H}_\mathsf{S} := \mathsf{H}_2 - \mathsf{A}_{12}^t\,\mathsf{Z}_{11}$
    $\mathsf{u}_\mathsf{S} := \mathsf{u}_{22} - \mathsf{A}_{12}^t\mathsf{A}_{11}^{-t}\mathsf{u}_{21}$
    if the $(1,1)$ element of S is zero then
      $(\ell, \mathsf{Y}, \mathsf{Z}, \mathsf{v}) := (\ell_{11}, \mathsf{Y}_{11}, \mathsf{Z}_{11}, \mathsf{v}_{11})$
    else
      $(\ell_\mathsf{S}, \mathsf{Y}_\mathsf{S}, \mathsf{Z}_\mathsf{S}, \mathsf{v}_\mathsf{S}) := \mathsf{largest\_rec}(\mathsf{G}_\mathsf{S}, \mathsf{H}_\mathsf{S}, \mathsf{u}_\mathsf{S})$
      $\ell := \ell_{11} + \ell_\mathsf{S}$
      $\mathsf{Y} := \begin{bmatrix} \mathsf{Y}_{11} - (\mathsf{A}_{11}^{-1}\mathsf{A}_{12})\mathsf{Y}_\mathsf{S} \\ \mathsf{Y}_\mathsf{S} \end{bmatrix}$; $\mathsf{Z} := \begin{bmatrix} \mathsf{Z}_{11} - (\mathsf{A}_{11}^{-T}\mathsf{A}_{21}^{T})\mathsf{Z}_\mathsf{S} \\ \mathsf{Z}_\mathsf{S} \end{bmatrix}$;
      $\mathsf{w} := -\mathsf{S}^{-T}\mathsf{A}_{12}^T\mathsf{v}_{11}$; $\mathsf{v} := \begin{bmatrix} \mathsf{v}_{11} - \mathsf{A}_{11}^{-T}\mathsf{A}_{21}^T\mathsf{w} \\ \mathsf{w} \end{bmatrix}$;
return $(\ell, \mathsf{Y}, \mathsf{Z}, \mathsf{v})$.

FIG. 4. *Algorithm* $\mathsf{largest\_rec}$. *Here* $\mathsf{A}_\ell \in \mathbb{F}^{\ell\times\ell}$ *denotes the largest leading principal submatrix of* A *whose rank profile is generic, and* G_ℓ *and* H_ℓ *are the* $\ell \times \alpha$ *submatrices consisting of the first* ℓ *rows of* G *and* H*, respectively.*

with $i_0 \in \mathbb{N}$ such that $\alpha \leqslant m/2^{i_0} < 2\alpha$, that is, $i_0 = \lfloor \log(m/\alpha) \rfloor$. Defining $\overline{\alpha} = 2^{\lceil \log(\alpha) \rceil}$, we have $\alpha \in (\overline{\alpha}/2, \overline{\alpha}]$, which implies $m/\alpha \in [m/\overline{\alpha}, 2m/\overline{\alpha})$ and thus, since m is an integer power of two,

$$2^{i_0} = m/\overline{\alpha}.$$

Besides, $m/(2^{i+1}\alpha) < m/(2^i\overline{\alpha})$, which by assumption on $\mathscr{M}'_{\mathrm{mat}}$ implies

$$\mathscr{M}'_{\mathrm{mat}}\Big(\frac{m}{2^{i+1}\alpha}, \alpha\Big) = O\Big(\mathscr{M}'_{\mathrm{mat}}\Big(\frac{m}{2^i\overline{\alpha}}, \alpha\Big)\Big).$$

Third, $\alpha^\omega = O\big(\mathscr{M}'_{\mathrm{mat}}(1,\alpha)\big)$. Consequently, for some constant c_1,

$$C(m,\alpha) \leqslant c_1 \cdot \sum_{i=0}^{\log(m/\overline{\alpha})} 2^i \mathscr{M}'_{\mathrm{mat}}\Big(\frac{m}{2^i\overline{\alpha}}, \alpha\Big).$$

Now, $\alpha \leqslant \overline{\alpha}$ implies $m/\overline{\alpha} \leqslant m/\alpha \leqslant \overline{m/\alpha}$, where $\overline{m/\alpha}$ denotes the smallest integer power of two greater than or equal to m/α. Hence $\log(m/\overline{\alpha}) \leqslant \log\big(\overline{m/\alpha}\big)$ and $\mathscr{M}'_{\mathrm{mat}}(2^{-i}m/\overline{\alpha}) = O\big(\mathscr{M}'_{\mathrm{mat}}\big(2^{-i}\overline{m/\alpha}\big)\big)$ by assumption on $\mathscr{M}'_{\mathrm{mat}}$, so that the sum in the above bound on $C(m,\alpha)$ is in $O(\mathscr{M}''_{\mathrm{mat}}(m/\alpha,\alpha))$, as announced. ∎

Algorithm $\mathsf{largest}(\mathsf{G}, \mathsf{H}, \mathsf{u})$

Input: $(\mathsf{G}, \mathsf{H}, \mathsf{u}) \in \mathbb{F}^{m\times\alpha} \times \mathbb{F}^{n\times\alpha} \times \mathbb{F}^n$ such that $\alpha \leqslant \min(m,n)$ and $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,0}}(\mathsf{A}) = \mathsf{G}\mathsf{H}^t$ and $\mathsf{u}^t = \mathsf{e}^t_{m,m}\mathsf{A}$ (the last row of A).

Output: $(\ell, \mathsf{Y}, \mathsf{Z}, \mathsf{v}) \in \mathbb{N} \times \mathbb{F}^{\ell\times\alpha} \times \mathbb{F}^{\ell\times\alpha} \times \mathbb{F}^\ell$ such that ℓ is the order of the largest leading principal submatrix A_ℓ of A having generic rank profile, $\mathsf{Y} = -\mathsf{A}_\ell^{-1}\mathsf{G}_\ell$, $\mathsf{Z} = \mathsf{A}_\ell^{-t}\mathsf{H}_\ell$ and $\mathsf{v} = \mathsf{A}_\ell^{-t}\mathsf{e}_{\ell,1}$ (the first row of A_ℓ^{-1}).

$\overline{p} := 2^{\lceil \log_2(\max(m,n))\rceil}$

compute $\overline{\alpha} \in \mathbb{N}$, $\overline{\mathsf{G}}, \overline{\mathsf{H}} \in \mathbb{F}^{\overline{p}\times\overline{\alpha}}$, and $\overline{\mathsf{u}} \in \mathbb{F}^{\overline{p}}$ such that:

- $(\overline{\mathsf{G}}, \overline{\mathsf{H}}, \overline{\mathsf{u}})$ is a $\nabla_{\mathbb{Z}_{\overline{p},0},\mathbb{Z}^t_{\overline{p},0}}$-generator of length $\overline{\alpha}$ of $\overline{\mathsf{A}} = \left[\begin{smallmatrix} \mathsf{A} & 0 \\ 0 & 0 \end{smallmatrix}\right] \in \mathbb{F}^{\overline{p}\times\overline{p}}$;
- $\alpha \leqslant \overline{\alpha} \leqslant \min(\alpha+2, \overline{p})$;
- $\overline{\mathsf{G}} = \left[\begin{smallmatrix} \mathsf{G} & * \\ * & * \end{smallmatrix}\right]$ and $\overline{\mathsf{H}} = \left[\begin{smallmatrix} \mathsf{H} & * \\ * & * \end{smallmatrix}\right]$
- $\overline{\mathsf{u}}^t$ is the last row of $\overline{\mathsf{A}}$

$(\ell, \overline{\mathsf{Y}}, \overline{\mathsf{Z}}, \mathsf{v}) := \mathsf{largest\_rec}(\overline{\mathsf{G}}, \overline{\mathsf{H}}, \overline{\mathsf{u}})$

$\mathsf{Y} := \overline{\mathsf{Y}}^{\mapsto\alpha}$; $\mathsf{Z} := \overline{\mathsf{Z}}^{\mapsto\alpha}$

return $(\ell, \mathsf{Y}, \mathsf{Z}, \mathsf{v})$.

FIG. 5. *Algorithm* largest. *Here* $\mathsf{A}_\ell \in \mathbb{F}^{\ell\times\ell}$ *denotes the largest leading principal submatrix of* A *whose rank profile is generic, and* G_ℓ *and* H_ℓ *are the* $\ell \times \alpha$ *submatrices consisting of the first* ℓ *rows of* G *and* H*, respectively.*

LEMMA 37. *Let* $p = \max(m,n)$. *Algorithm* largest *works correctly in time*

$$O\Big(\mathscr{M}''_{\mathrm{mat}}\Big(\frac{p}{\alpha}, \alpha\Big)\Big).$$

Proof. Let us show first how to generate the augmented matrix $\overline{\mathsf{A}}$. If $\overline{p} = m = n$, then $\overline{\mathsf{A}} = \mathsf{A}$ and it suffices to take $\overline{\alpha} = \alpha$, $\overline{\mathsf{G}} = \mathsf{G}$, and $\overline{\mathsf{H}} = \mathsf{H}$. If $\overline{p} = n > m$ then $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,0}}(\mathsf{A}) = \mathsf{G}\mathsf{H}^t$ leads to the following generator of length $\overline{\alpha} := \alpha + 1$:

$$\nabla_{\mathbb{Z}_{\overline{p},0},\mathbb{Z}^t_{\overline{p},0}}(\overline{\mathsf{A}}) = \overline{\mathsf{G}}\,\overline{\mathsf{H}}^t, \qquad \overline{\mathsf{G}} = \begin{bmatrix} \mathsf{G} & \\ & \mathsf{e}_{\overline{p}-m,1} \end{bmatrix}, \qquad \overline{\mathsf{H}} = \big[\,\mathsf{H} \,|\, \mathsf{u}\,\big];$$

the range constraint on $\overline{\alpha}$ is satisfied, since $\alpha \leqslant \min(m,n) = m < n = \overline{p}$, and, on the other hand, the upper left corners of $\overline{\mathsf{G}}$ and $\overline{\mathsf{H}}$ are G and H, respectively. Proceeding similarly when $\overline{p} = m > n$, we take

$$\overline{\alpha} = \alpha + 1, \qquad \overline{\mathsf{G}} = \big[\,\mathsf{G} \,|\, -\mathsf{u}'\,\big], \qquad \overline{\mathsf{H}} = \begin{bmatrix} \mathsf{H} & \\ & \mathsf{e}_{\overline{p}-n,1} \end{bmatrix},$$

where u' is the last column of A and can be obtained in time $O(\alpha\mathsf{M}(\overline{p}))$. It remains to handle the case $\overline{p} > \max(m,n)$, where A is bordered by both some zero rows and some zero columns. In this case, we deduce from $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,0}}(\mathsf{A}) = \mathsf{G}\mathsf{H}^t$ that a $\nabla_{\mathbb{Z}_{\overline{p},0},\mathbb{Z}^t_{\overline{p},0}}$-generator of length $\alpha + 2$ of $\overline{\mathsf{A}}$ is given by

$$\overline{\mathsf{G}} = \begin{bmatrix} \mathsf{G} & & -\mathsf{u}' \\ & \mathsf{e}_{\overline{p}-m,1} & \end{bmatrix}, \qquad \overline{\mathsf{H}} = \begin{bmatrix} \mathsf{H} & \mathsf{u} & \\ & & \mathsf{e}_{\overline{p}-n,1} \end{bmatrix}.$$

Again, it is clear that the upper left corners of $\overline{\mathsf{G}}$ and $\overline{\mathsf{H}}$ are G and H, respectively. Furthermore, if $\overline{p} \geqslant \alpha + 2$, we can take $\overline{\alpha} = \alpha + 2$; otherwise, since $\overline{p} > \max(m,n) \geqslant$

$\min(m,n) \geqslant \alpha$, we are in the situation where $\overline{p} = \alpha + 1$ and $m = n = \alpha$. In this case we shall proceed as follows to reduce the generator length from $\alpha + 2$ to $\alpha + 1$ while ensuring that G and H are in the upper left corners of the new generator matrices: the matrices $\overline{\mathsf{G}}$ and $\overline{\mathsf{H}}$ defined above have dimensions $(\alpha+1) \times (\alpha+2)$ and are given by

$$\overline{\mathsf{G}} = \begin{bmatrix} \mathsf{G} & & -\mathsf{u}' \\ & 1 & \end{bmatrix} \qquad \text{and} \qquad \overline{\mathsf{H}}^t = \begin{bmatrix} \mathsf{H}^t & & \\ \mathsf{u}^t & & \\ & & 1 \end{bmatrix};$$

since G is $\alpha \times \alpha$, we can deduce from G and u' a vector $\mathsf{u}'' \in \mathbb{F}^\alpha$ such that $\mathsf{G}\mathsf{u}'' = \mathsf{u}'$ in time $O(\alpha^\omega)$; then, using the fact that

$$\mathsf{E} = \begin{bmatrix} \mathbb{I}_\alpha & & \mathsf{u}'' \\ & 1 & \\ & & 1 \end{bmatrix} \qquad \Longrightarrow \qquad \overline{\mathsf{G}}\mathsf{E} = \begin{bmatrix} \mathsf{G} & & 0 \\ & 1 & 0 \end{bmatrix} \quad \text{and} \quad \mathsf{E}^{-1}\overline{\mathsf{H}}^t = \begin{bmatrix} \mathsf{H}^t & -\mathsf{u}'' \\ \mathsf{u}^t & \\ & 1 \end{bmatrix},$$

we conclude that a suitable $\nabla_{\mathbb{Z}_{\overline{p},0},\mathbb{Z}^t_{\overline{p},0}}$-generator of $\overline{\mathsf{A}}$ is obtained by taking the first $\alpha + 1 =: \overline{\alpha}$ columns of $\overline{\mathsf{G}}\mathsf{E}$ and $\overline{\mathsf{H}}\mathsf{E}^{-t}$. Finally, obtaining the vector $\overline{\mathsf{u}}$ is free, since the last row of $\overline{\mathsf{A}}$ is either the last row of A (which is part of the input) or the zero row. To summarize, we can always find a suitable generator $(\overline{\mathsf{G}}, \overline{\mathsf{H}}, \overline{\mathsf{u}})$ of the augmented matrix $\overline{\mathsf{A}}$ in time $O(\alpha \mathsf{M}(\overline{p}) + \alpha^\omega)$, that is, since $\overline{p} < 2p$ and $\alpha \leqslant p$,

$$O(\alpha\, \mathsf{M}(p) + \alpha^{\omega-1} p).$$

Since $\overline{\alpha} \leqslant \overline{p}$ and $\overline{p}$ is a power of two, Lemma 36 implies that the output $(\ell, \mathsf{Y}, \mathsf{Z}, \mathsf{v})$ of largest_rec is obtained in time

$$O(\mathscr{M}''_{\mathrm{mat}}(\overline{p}/\overline{\alpha}, \overline{\alpha}))$$

and satisfies the following: ℓ is the size of the largest leading principal submatrix of A having generic rank profile; furthermore, denoting this matrix by A_ℓ, we have v^t equal to the first row of A_ℓ^{-1} and $\overline{\mathsf{Y}} = -\mathsf{A}_\ell^{-1}\overline{\mathsf{G}}_\ell$ and $\overline{\mathsf{Z}} = -\mathsf{A}_\ell^{-t}\overline{\mathsf{H}}_\ell$ with $\overline{\mathsf{G}}_\ell$ and $\overline{\mathsf{H}}_\ell$ the $\ell \times \overline{\alpha}$ submatrices consisting of the first ℓ rows of $\overline{\mathsf{G}}$ and $\overline{\mathsf{H}}$. Since $\mathsf{G} \in \mathbb{F}^{m\times\alpha}$ and $\mathsf{H} \in \mathbb{F}^{n\times\alpha}$ are in the upper left corners of $\overline{\mathsf{G}}$ and $\overline{\mathsf{H}}$ and since $\ell \leqslant \min(m,n)$, we deduce that

$$\overline{\mathsf{G}}_\ell = [\mathsf{G}_\ell \ *] \qquad \text{and} \qquad \overline{\mathsf{H}}_\ell = [\mathsf{H}_\ell \ *].$$

Consequently, by keeping only the first α columns of each of $\overline{\mathsf{Y}}$ and $\overline{\mathsf{Z}}$, we obtain the matrix pair (Y, Z) such that

$$\mathsf{Y} = -\mathsf{A}_\ell^{-1}\mathsf{G}_\ell \qquad \text{and} \qquad \mathsf{Z} = -\mathsf{A}_\ell^{-t}\mathsf{H}_\ell.$$

This concludes the proof of correctness. For the cost, note that the smallest integer power of two greater than or equal to $\overline{p}/\overline{\alpha}$ is less than $4p/\alpha$ and, on the other hand, and that $\overline{\alpha} = O(\alpha)$. Hence $\mathscr{M}''_{\mathrm{mat}}(\overline{p}/\overline{\alpha}, \overline{\alpha}) = O\big(\mathscr{M}''_{\mathrm{mat}}(p/\alpha, \alpha)\big)$, and the latter expression dominates over the term in $O(\alpha\, \mathsf{M}(p) + \alpha^{\omega-1} p)$. ∎

LEMMA 38. *Let $p = \max(m,n)$. Algorithm* lp_inv *works correctly in time*

$$O\left(\mathscr{M}''_{\mathrm{mat}}\left(\frac{p}{\alpha}, \alpha\right)\right).$$

Algorithm $\mathsf{lp\_inv}(\mathsf{G}, \mathsf{H}, \mathsf{u})$

Input: $(\mathsf{G}, \mathsf{H}) \in \mathbb{F}^{m\times\alpha} \times \mathbb{F}^{n\times\alpha}$ such that $\alpha \leqslant \min(m,n)$ and $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}^t_{n,0}}(\mathsf{A}) = \mathsf{G}\mathsf{H}^t$;
$\mathsf{u} \in \mathbb{F}^n$ such that $\mathsf{u}^t = \mathsf{e}^t_{m,m}\mathsf{A}$ (the last row of A).
Output: $(r, \mathsf{Y}, \mathsf{Z}, \mathsf{v})$ such that $r = \operatorname{rank}(\mathsf{A})$, $(\mathsf{Y}, \mathsf{Z}) = (-\mathsf{A}_r^{-1}\mathsf{G}_r, \mathsf{A}_r^{-t}\mathsf{H}_r)$, and $\mathsf{v} = \mathsf{A}_r^{-t}\mathsf{e}_{r,1}$ (the first row of A_r^{-1}) if A has generic rank profile, and $r =$ "failure" otherwise.

$(\ell, \mathsf{Y}_\ell, \mathsf{Z}_\ell, \mathsf{v}_\ell) := \mathsf{largest}(\mathsf{G}, \mathsf{H}, \mathsf{u})$
$\mathsf{G}_\mathsf{S} := \mathsf{G}_2 + \mathsf{A}_{21}\mathsf{Y}_\ell$; $\mathsf{H}_\mathsf{S} := \mathsf{H}_2 - \mathsf{A}_{12}^t\,\mathsf{Z}_\ell$
$\mathsf{u}_\mathsf{S} := \mathsf{u}_{22} - \mathsf{A}_{12}^t\mathsf{A}_\ell^{-t}\mathsf{u}_{21}$
$r_0 :=$ the rank of $[\mathsf{G}_\mathsf{S}|\mathsf{e}_{m-\ell,1}]\,[\mathsf{H}_\mathsf{S}|\mathsf{u}_\mathsf{S}]^t$
if $r_0 = 0$ then
    $(r, \mathsf{Y}, \mathsf{Z}, \mathsf{v}) := (\ell, \mathsf{Y}_\ell, \mathsf{Z}_\ell, \mathsf{v}_\ell)$
else
    $(r, \mathsf{Y}, \mathsf{Z}, \mathsf{v}) := (\text{"failure"}, *, *, *)$
return $(r, \mathsf{Y}, \mathsf{Z}, \mathsf{v})$.

FIG. 6. *Algorithm* $\mathsf{lp\_inv}$. *Here* r *denotes the (unknown) rank of* A, *and* A_r *is the* $r \times r$ *matrix consisting of the first* r *rows and columns of* A. *Similarly,* G_r *and* H_r *are the* $r\times\alpha$ *matrices consisting of the first* r *rows of* G *and* H, *respectively.*

Proof. By Lemma 37, the call to $\mathsf{largest}$ yields $(\ell, \mathsf{Y}_\ell, \mathsf{Z}_\ell, \mathsf{v}_\ell)$ such that ℓ is the order of the largest leading principal submatrix of A having generic rank profile, $\mathsf{Y}_\ell = -\mathsf{A}_\ell^{-1}\mathsf{G}_\ell$, $\mathsf{Z}_\ell = \mathsf{A}_\ell^{-t}\mathsf{H}_\ell$, and $\mathsf{v}_\ell = \mathsf{A}_\ell^{-t}\mathsf{e}_{\ell,1}$. Thanks to the special shape of the matrices Y_ℓ and Z_ℓ, we can check as in [16] that the expressions for G_S, H_S, u_S lead to a $\nabla_{\mathbb{Z}_{m-\ell,1},\mathbb{Z}^t_{n-\ell,0}}$-generator of length $\alpha+1$ of the Schur complement $\mathsf{S} = \mathsf{A}_{22} - \mathsf{A}_{21}\mathsf{A}_\ell^{-1}\mathsf{A}_{12}$:

$$\nabla_{\mathbb{Z}_{m-\ell,1},\mathbb{Z}^t_{n-\ell,0}}(\mathsf{S}) = [\mathsf{G}_\mathsf{S}|\mathsf{e}_{m-\ell,1}][\mathsf{H}_\mathsf{S}|\mathsf{u}_\mathsf{S}]^t.$$

Now, since both $\nabla_{\mathbb{Z}_{m-\ell,1},\mathbb{Z}^t_{n-\ell,0}}$ and A_ℓ are invertible,

$$r_0 := \operatorname{rank}\left(\nabla_{\mathbb{Z}_{m-\ell,1},\mathbb{Z}^t_{n-\ell,0}}(\mathsf{S})\right) = 0 \quad\Longleftrightarrow\quad \mathsf{S} = 0 \quad\Longleftrightarrow\quad \ell = \operatorname{rank}(\mathsf{A}).$$

Correctness then follows, since $\ell = \operatorname{rank}(\mathsf{A})$ if and only if A has generic rank profile.

To bound the cost, recall first from Lemma 37 than ℓ, Y_ℓ, Z_ℓ, v_ℓ are computed in time $O(\mathscr{M}''_{\mathrm{mat}}(p/\alpha,\alpha))$. Then we can obtain G_S in time $O(\mathscr{M}'_{\mathrm{mat}}(p/\alpha,\alpha))$ by evaluating the product $\mathsf{A}_{21}\mathsf{Y}_\ell$ as follows. Defining $\overline{\mathsf{A}}_{21} = \left[\begin{smallmatrix} 0 & 0 \\ 0 & \mathsf{A}_{21}\end{smallmatrix}\right] \in \mathbb{F}^{m\times n}$ and $\overline{\mathsf{Y}}_\ell = \left[\begin{smallmatrix} 0 \\ \mathsf{Y}_\ell\end{smallmatrix}\right] \in \mathbb{F}^{n\times\alpha}$, we can deduce from $\mathsf{G}, \mathsf{H}, \mathsf{u}$ a $\nabla_{\mathbb{Z}_{m,1},\mathbb{Z}_{n,0}}$-generator $(\overline{\mathsf{G}}_{21}, \overline{\mathsf{H}}_{21})$ of length $\alpha+2$ of $\overline{\mathsf{A}}_{21}$ in time $O(\alpha\,\mathsf{M}(p))$. Writing $\overline{\mathsf{A}}_{21} = \overline{\mathsf{A}}'_{21} + \overline{\mathsf{A}}''_{21}$ with $\overline{\mathsf{A}}'_{21}$ of displacement rank $\alpha \leqslant \min(m,n)$ and $\overline{\mathsf{A}}''_{21}$ of displacement rank 2, we can evaluate $\overline{\mathsf{A}}_{21}\overline{\mathsf{Y}}_\ell$ in time $\mathscr{C}_{\mathrm{mul}}(\nabla_{\mathbb{Z}_{m,1},\mathbb{Z}^t_{n,0}}, \alpha, \alpha) + O(\alpha\,\mathsf{M}(p))$, which by Theorem 2 is in $O(\mathscr{M}'_{\mathrm{mat}}(p/\alpha,\alpha) + \alpha\,\mathsf{M}(p))$. It remains to extract $\mathsf{A}_{21}\mathsf{Y}_\ell$ from $\overline{\mathsf{A}}_{21}\overline{\mathsf{Y}}_\ell = \left[\begin{smallmatrix} 0 \\ \mathsf{A}_{21}\mathsf{Y}_\ell\end{smallmatrix}\right]$ and to add it to G_2, for an overhead of $O(\alpha p)$. Since $\alpha\,\mathsf{M}(p) = O(\mathscr{M}'_{\mathrm{mat}}(p/\alpha,\alpha))$, we have thus obtained G_S in time $O(\mathscr{M}'_{\mathrm{mat}}(p/\alpha,\alpha))$. The same cost bound can be derived for the matrix H_S and, on the other hand, the cost bound $O(\alpha\,\mathsf{M}(p))$ applies to the computation of u_S and follows from computing a generator of length $O(\alpha)$ of $\mathsf{A}_{12}^t\mathsf{A}_\ell^{-t}$ and then multiplying by and subtracting from a vector. Finally, given G_S H_S, u_S, the rank r_0 can be deduced in time $O(\alpha^{\omega-1}p)$. The conclusion for the cost then follows from the fact that $\alpha^{\omega-1}p$ and $\mathscr{M}'_{\mathrm{mat}}(p/\alpha,\alpha)$ are both in $O(\mathscr{M}''_{\mathrm{mat}}(p/\alpha,\alpha))$. ∎

6.3. Generating inverses and solving linear systems. We conclude by showing how to apply algorithm lp_inv from the previous section to our initial problems $\mathsf{inv}(\mathcal{L},\alpha)$ and $\mathsf{solve}(\mathcal{L},\alpha)$. This corresponds to algorithms inv and solve given in Figures 7 and 8.

Algorithm $\mathsf{inv}(\mathsf{G},\mathsf{H},S)$

Input: $(\mathsf{G},\mathsf{H}) \in \mathbb{F}^{m\times\alpha}\times\mathbb{F}^{m\times\alpha}$ such that $\alpha \leqslant m$ and $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{m,1}^t}(\mathsf{A}) = \mathsf{G}\mathsf{H}^t$, and a finite subset $S \subset \mathbb{F}$.
Output: if not "failure", then $(\mathsf{Y},\mathsf{Z}) = (-\mathsf{A}^{-1}\mathsf{G}, \mathsf{A}^{-t}\mathsf{H})$ or "A is singular".

choose the entries of $\mathsf{r}_1,\mathsf{r}_2 \in \mathbb{F}^{m-1}$ uniformly at random from S
$\mathsf{v}_1 := \left[1 \mid \mathsf{r}_1^t\right]^t$; $\mathsf{v}_2 := \left[1 \mid \mathsf{r}_2^t\right]^t$
$(\widetilde{\mathsf{G}},\widetilde{\mathsf{H}},\widetilde{\mathsf{u}}) := \mathsf{precond}(\mathsf{G},\mathsf{H},\mathsf{v}_1,\mathsf{v}_2)$
$(r,\widetilde{\mathsf{Y}},\widetilde{\mathsf{Z}},*) := \mathsf{lp\_inv}(\widetilde{\mathsf{G}},\widetilde{\mathsf{H}},\widetilde{\mathsf{u}})$
if $r \notin \{0,1,\ldots,m\}$ then
    return "failure"
else
    if $r = m$ then
        $\mathsf{Y} := \mathbb{U}(\mathsf{v}_2)^t\,\widetilde{\mathsf{Y}}^{\mapsto\alpha}$; $\mathsf{Z} := \mathbb{U}(\mathsf{v}_1)^t\,\widetilde{\mathsf{Z}}^{\mapsto\alpha}$
        return (Y,Z)
    else
        return "A is singular"

FIG. 7. *Algorithm* inv.

THEOREM 39. *Algorithm* inv *works correctly in time* $O\left(\mathscr{M}''_{\mathrm{mat}}\left(\frac{m}{\alpha},\alpha\right)\right)$. *It makes* $2m-2$ *random choices in* $\mathbb{F}$ *and fails with probability less than* $1/2$ *if* $|S| \geqslant 2m(m+1)$.

Proof. By applying Lemma 35 to the case $m = n$, we see that $\widetilde{\mathsf{u}}^t$ is the last row of $\widetilde{\mathsf{A}} = \mathbb{U}(\mathsf{v}_1)\mathsf{A}\mathbb{U}(\mathsf{v}_2)^t$ and that $\widetilde{\mathsf{G}}$ and $\widetilde{\mathsf{H}}$ are $m\times(\alpha+4)$ matrices such that

$$\mathbb{Z}_{m,0}\,\widetilde{\mathsf{A}} - \widetilde{\mathsf{A}}\,\mathbb{Z}_{m,0}^t = \widetilde{\mathsf{G}}\widetilde{\mathsf{H}}^t, \qquad \widetilde{\mathsf{G}}^{\mapsto\alpha} = \mathbb{U}(\mathsf{v}_1)\mathsf{G}, \qquad \widetilde{\mathsf{H}}^{\mapsto\alpha} = \mathbb{U}(\mathsf{v}_2)\mathsf{H}.$$

If $r \notin \{0,1,\ldots,m\}$ then $r =$ "failure", and this is what we return. Assume now that $r \in \{0,1,\ldots,m\}$. In this case, preconditioning has ensured that $\widetilde{\mathsf{A}}$ has generic rank profile. The number r produced by lp_inv thus satisfies $r = \mathrm{rank}(\widetilde{\mathsf{A}}) = \mathrm{rank}(\mathsf{A})$, and A is singular if and only if $r \neq m$. When $r = m$, we have $\widetilde{\mathsf{A}} = \widetilde{\mathsf{A}}_r$, so the matrices $\widetilde{\mathsf{Y}}$ and $\widetilde{\mathsf{Z}}$ produced by lp_inv satisfy $\widetilde{\mathsf{Y}} = -\widetilde{\mathsf{A}}^{-1}\widetilde{\mathsf{G}}$ and $\widetilde{\mathsf{Z}} = \widetilde{\mathsf{A}}^{-t}\widetilde{\mathsf{H}}$. Using the special shape of the first α columns of $\widetilde{\mathsf{G}}$ and $\widetilde{\mathsf{H}}$, we conclude that the returned matrices Y and Z are $-\mathsf{A}^{-1}\mathsf{G}$ and $\mathsf{A}^{-t}\mathsf{H}$, as wanted. (Note that although it is produced by lp_inv, the first row of the inverse of $\widetilde{\mathsf{A}}_r$ is not needed here and denoted by $*$; we will need it, however, when solving linear systems at the end of this section.)

Applying Lemmas 35 and 38 with $p = m$ shows that the cost of calling precond and lp_inv is $O(\alpha\,\mathsf{M}(m) + \mathscr{M}''_{\mathrm{mat}}(m/\alpha,\alpha))$; on the other hand, the Toeplitz structure of $\mathbb{U}(\mathsf{v}_1)$ and $\mathbb{U}(\mathsf{v}_2)$ implies that Y and Z can be deduced in time $O(\alpha\,\mathsf{M}(m))$ from v_1, v_2, $\widetilde{\mathsf{Y}}$, and $\widetilde{\mathsf{Z}}$. Writing $\overline{\alpha}$ and $\overline{m/\alpha}$ for the smallest integer powers of two greater than or equal to α and m/α, we have $\mathscr{M}''_{\mathrm{mat}}(m/\alpha,\alpha) \geqslant \mathscr{M}'_{\mathrm{mat}}\big(\overline{m/\alpha},\alpha\big) \geqslant \overline{\alpha}\,\mathscr{M}_{\mathrm{mat}}\big(\overline{\alpha}\cdot\overline{m/\alpha},1\big) = \overline{\alpha}\,\mathsf{M}\big(\overline{\alpha}\cdot\overline{m/\alpha}\big) \geqslant \alpha\,\mathsf{M}(m)$, from which the claimed cost bound follows.

Finally, by Lemma 35, the preconditioned matrix $\widetilde{\mathsf{A}}$ has generic rank profile with probability $\mathcal{P} \geqslant 1 - \operatorname{rank}(\mathsf{A}) \cdot (\operatorname{rank}(\mathsf{A}) + 1)/|S|$. Since $\operatorname{rank}(\mathsf{A}) \leqslant m$, we have $\mathcal{P} \geqslant 1 - m(m+1)/|S|$, so that $|S| \geqslant 2m(m+1)$ implies $\mathcal{P} \geqslant 1/2$. ∎

For solving $\mathsf{A}\mathsf{x} = \mathsf{b}$, we work on the equivalent (preconditioned) system $\widetilde{\mathsf{A}}\widetilde{\mathsf{x}} = \widetilde{\mathsf{b}}$ such that $\widetilde{\mathsf{A}} = \mathbb{U}(\mathsf{v}_1)\mathsf{A}\mathbb{U}(\mathsf{v}_2)^t$ and $\widetilde{\mathsf{b}} = \mathbb{U}(\mathsf{v}_1)\mathsf{b}$, for which any solution $\widetilde{\mathsf{x}}$ yields a solution $\mathsf{x} = \mathbb{U}(\mathsf{v}_2)^t\widetilde{\mathsf{x}}$. Algorithm solve in Figure 8 uses the following notation: writing as before r for the rank of A, we partition $\widetilde{\mathsf{A}}$ and $\widetilde{\mathsf{b}}$ into blocks as

$$\widetilde{\mathsf{A}} = \begin{bmatrix} \widetilde{\mathsf{A}}_r & \widetilde{\mathsf{A}}_{12} \\ \widetilde{\mathsf{A}}_{21} & \widetilde{\mathsf{A}}_{21}\widetilde{\mathsf{A}}_r^{-1}\widetilde{\mathsf{A}}_{12} \end{bmatrix}, \qquad \widetilde{\mathsf{b}} = \begin{bmatrix} \widetilde{\mathsf{b}}_1 \\ \widetilde{\mathsf{b}}_2 \end{bmatrix}$$

with $\mathsf{A}_{12} \in \mathbb{F}^{r\times(n-r)}$, $\widetilde{\mathsf{A}}_{21} \in \mathbb{F}^{(m-r)\times r}$, $\widetilde{\mathsf{b}}_1 \in \mathbb{F}^r$ and $\widetilde{\mathsf{b}}_2 \in \mathbb{F}^{m-r}$.

Algorithm $\mathsf{solve}(\mathsf{G}, \mathsf{H}, \mathsf{b}, S)$

Input: $(\mathsf{G}, \mathsf{H}) \in \mathbb{F}^{m\times\alpha} \times \mathbb{F}^{n\times\alpha}$ such that $\alpha \leqslant \min(m,n)$ and $\nabla_{\mathbb{Z}_{m,0},\mathbb{Z}_{n,1}^t}(\mathsf{A}) = \mathsf{G}\mathsf{H}^t$, $\mathsf{b} \in \mathbb{F}^m$, and a finite subset $S \subset \mathbb{F}$.
Output: if not "failure", then a nontrivial solution $\mathsf{x} \in \mathbb{F}^n$ to $\mathsf{A}\mathsf{x} = \mathsf{b}$ or "no solution exists."

choose the entries of $\mathsf{r}_1 \in \mathbb{F}^{m-1}$ and $\mathsf{r}_2 \in \mathbb{F}^{n-1}$ uniformly at random from S
$\mathsf{v}_1 := \left[1 \mid \mathsf{r}_1^t\right]^t$; $\mathsf{v}_2 := \left[1 \mid \mathsf{r}_2^t\right]^t$
$(\widetilde{\mathsf{G}}, \widetilde{\mathsf{H}}, \widetilde{\mathsf{u}}) := \mathsf{precond}(\mathsf{G}, \mathsf{H}, \mathsf{v}_1, \mathsf{v}_2)$
$(r, \widetilde{\mathsf{Y}}, \widetilde{\mathsf{Z}}, \widetilde{\mathsf{v}}) := \mathsf{lp\_inv}(\widetilde{\mathsf{G}}, \widetilde{\mathsf{H}}, \widetilde{\mathsf{u}})$
if $r \notin \{0, 1, \ldots, m\}$ then
    return "failure"
else
    $\begin{bmatrix} \widetilde{\mathsf{b}}_1 \\ \widetilde{\mathsf{b}}_2 \end{bmatrix} := \mathbb{U}(\mathsf{v}_1)\mathsf{b}$
    $\widetilde{\mathsf{x}}_1 := \widetilde{\mathsf{A}}_r^{-1}\widetilde{\mathsf{b}}_1$
    if $\widetilde{\mathsf{A}}_{21}\widetilde{\mathsf{x}}_1 = \widetilde{\mathsf{b}}_2$ then
        $\widetilde{\mathsf{x}} := \begin{bmatrix} \widetilde{\mathsf{x}}_1 + \widetilde{\mathsf{A}}_r^{-1}\widetilde{\mathsf{A}}_{12}\,\mathsf{e}_{n-r,1} \\ -\mathsf{e}_{n-r,1} \end{bmatrix}$
        $\mathsf{x} := \mathbb{U}(\mathsf{v}_2)^t\widetilde{\mathsf{x}}$
        return x
    else
        return "no solution exists"

FIG. 8. *Algorithm* solve.

THEOREM 40. *Algorithm* solve *works correctly in time* $O\left(\mathscr{M}''_{\mathrm{mat}}\left(\frac{p}{\alpha}, \alpha\right)\right)$ *with* $p = \max(m,n)$. *It makes* $m + n - 2$ *random choices in* $\mathbb{F}$ *and fails with probability less than* $1/2$ *if* $|S| \geqslant 2q(q+1)$ *with* $q = \min(m,n)$.

Proof. Recalling that $\mathsf{A}\mathsf{x} = \mathsf{b}$ is equivalent to $\widetilde{\mathsf{A}}\widetilde{\mathsf{x}} = \widetilde{\mathsf{b}}$ with $\widetilde{\mathsf{A}} = \mathbb{U}(\mathsf{v}_1)\mathsf{A}\mathbb{U}(\mathsf{v}_2)^t$, $\widetilde{\mathsf{x}} = \mathbb{U}(\mathsf{v}_2)^{-t}\mathsf{x}$ and $\widetilde{\mathsf{b}} = \mathbb{U}(\mathsf{v}_1)\mathsf{b}$, we see that the algorithm begins by generating the matrix $\widetilde{\mathsf{A}}$. If $r \in \{0, 1, \ldots, m\}$, then $\widetilde{\mathsf{A}}$ has generic rank profile and thus $r = \operatorname{rank}(\widetilde{\mathsf{A}}) = \operatorname{rank}(\mathsf{A})$. This implies $\widetilde{\mathsf{A}}_{22} - \widetilde{\mathsf{A}}_{21}\widetilde{\mathsf{A}}_r^{-1}\widetilde{\mathsf{A}}_{12} = 0$ and, partitioning $\widetilde{\mathsf{b}}$ conformally with $\widetilde{\mathsf{A}}$,

we deduce that the linear system $\widetilde{\mathsf{A}}\,\widetilde{\mathsf{x}} = \widetilde{\mathsf{b}}$ is equivalent to

$$\begin{bmatrix} \widetilde{\mathsf{A}}_r & \widetilde{\mathsf{A}}_{12} \\ 0 & 0 \end{bmatrix} \widetilde{\mathsf{x}} = \begin{bmatrix} \mathbb{I}_r & 0 \\ -\widetilde{\mathsf{A}}_{21}\widetilde{\mathsf{A}}_r^{-1} & \mathbb{I}_{m-r} \end{bmatrix} \begin{bmatrix} \widetilde{\mathsf{b}}_1 \\ \widetilde{\mathsf{b}}_2 \end{bmatrix}.$$

If $\widetilde{\mathsf{b}}_2$ is such that $-\widetilde{\mathsf{A}}_{21}\widetilde{\mathsf{x}}_1 + \widetilde{\mathsf{b}}_2 \neq 0$ for $\widetilde{\mathsf{x}}_1 = \widetilde{\mathsf{A}}_r^{-1}\widetilde{\mathsf{b}}_1$, then no solution exists. Else, it is easily checked that any vector of the form

$$\widetilde{\mathsf{x}}(\mathsf{v}) = \begin{bmatrix} \widetilde{\mathsf{x}}_1 + \widetilde{\mathsf{A}}_r^{-1}\widetilde{\mathsf{A}}_{12}\mathsf{v} \\ -\mathsf{v} \end{bmatrix} \quad \text{with} \quad \mathsf{v} \in \mathbb{F}^{n-r}$$

is a solution to $\widetilde{\mathsf{A}}\,\widetilde{\mathsf{x}} = \widetilde{\mathsf{b}}$; furthermore, taking $\mathsf{v} = e_{n-r,1} \neq 0$ when $n - r > 0$ ensures that this solution is nontrivial. Pre-multiplying this solution by $\mathbb{U}(\mathsf{v}_2)^t$ then yields a nontrivial solution to the original system, so correctness follows.

By Lemmas 35 and 38, calling $\mathsf{precond}$ and $\mathsf{lp\_inv}$ uses $O(\alpha\,\mathsf{M}(p) + \mathscr{M}''_{\mathrm{mat}}(p/\alpha, \alpha))$ operations in $\mathbb{F}$. Then, since $\mathbb{U}(\mathsf{v}_1)$ and $\mathbb{U}(\mathsf{v}_2)$ are Toeplitz matrices, we can deduce $\widetilde{\mathsf{b}} = \mathbb{U}(\mathsf{v}_1)\mathsf{b}$ and $\mathsf{x} = \mathbb{U}(\mathsf{v}_2)^t\,\widetilde{\mathsf{x}}$ from v_1, v_2, b, $\widetilde{\mathsf{x}}$ in time $O(\mathsf{M}(p))$. Finally, given the generators $(\widetilde{\mathsf{G}}, \widetilde{\mathsf{H}}, \widetilde{\mathsf{v}})$ and $(\widetilde{\mathsf{Y}}, \widetilde{\mathsf{Z}}, \widetilde{\mathsf{v}})$ of $\widetilde{\mathsf{A}}$ and $\widetilde{\mathsf{A}}_r^{-1}$, we can generate the submatrices $\widetilde{\mathsf{A}}_{12}$ and $\widetilde{\mathsf{A}}_{21}$ and perform the matrix-vector products $\widetilde{\mathsf{A}}_r^{-1}\widetilde{\mathsf{b}}_1$, $\widetilde{\mathsf{A}}_{21}\widetilde{\mathsf{x}}_1$, and $\widetilde{\mathsf{A}}_r^{-1}\widetilde{\mathsf{A}}_{12}\,\mathsf{e}_{n-r,1}$ in time $O(\alpha\,\mathsf{M}(p))$. Recalling that $\alpha\mathsf{M}(p) \leqslant \mathscr{M}''_{\mathrm{mat}}(p/\alpha, \alpha)$, we conclude that the total cost is thus in $O(\mathscr{M}''_{\mathrm{mat}}(p/\alpha, \alpha))$.

The probability analysis is the same as for inversion. ∎

Remark. If one wants to sample uniformly the solution manifold of $\mathsf{A}\,\mathsf{x} = \mathsf{b}$, then it suffices to replace the unit vector $\mathsf{e}_{n-r,1}$ in algorithm solve by a vector $\mathsf{r}_3 \in \mathbb{F}^{n-r}$ whose entries are selected uniformly at random from the subset S. This way of constructing x corresponds to the technique introduced by Kaltofen and Saunders in [21, Theorem 4], but requires only $n - r$ additional random choices instead of n.

REFERENCES

[1] A. V. Aho, K. Steiglitz, and J. D. Ullman, *Evaluating polynomials at fixed sets of points*, SIAM J. Comput., 4 (1975), pp. 533–539.

[2] D. Bini and V. Y. Pan, *Polynomial and Matrix Computations, volume 1: Fundamental Algorithms*, Birkhäuser, 1994.

[3] R. R. Bitmead and B. D. O. Anderson, *Asymptotically fast solution of Toeplitz and related systems of linear equations*, Linear Algebra Appl., 34 (1980), pp. 103–116.

[4] A. Bostan, C.-P. Jeannerod, and É. Schost, *Solving structured linear systems with large displacement rank*, Theoret. Comput. Sci., 407 (2008), pp. 155–181.

[5] A. Bostan, G. Lecerf, B. Salvy, É. Schost, and B. Wiebelt, *Complexity issues in bivariate polynomial factorization*, Proceedings of ISSAC'04, ACM, 2004, pp. 42–49.

[6] A. Bostan, G. Lecerf, and É. Schost, *Tellegen's principle into practice*, Proceedings of ISSAC'03, ACM, 2003, pp. 37–44.

[7] A. Bostan and É. Schost, *Polynomial evaluation and interpolation on special sets of points*, J. Complexity, 21 (2005), pp. 420–446.

[8] P. Bürgisser, M. Clausen, and A. Shokrollahi, *Algebraic Complexity Theory*, Springer, 1997.

[9] D. G. Cantor and E. Kaltofen, *On fast multiplication of polynomials over arbitrary algebras*, Acta Inform., 28 (1991), pp. 693–701.

[10] M. F. I. Chowdhury, C.-P. Jeannerod, V. Neiger, É. Schost, and G. Villard, *Faster algorithms for multivariate interpolation with multiplicities and simultaneous polynomial approximations*, IEEE Trans. Inform. Theory, 61 (2015), pp. 2370–2387.

[11] B. Friedlander, M. Morf, T. Kailath, and L. Ljung, *New inversion formulas for matrices classified in terms of their distance from Toeplitz matrices*, Linear Algebra Appl., 27 (1979), pp. 31–60.

[12] J. von zur Gathen and J. Gerhard, *Modern computer algebra*, Cambridge University Press, third ed., 2013.
[13] I. Gohberg and V. Olshevsky, *Complexity of multiplication with vectors for structured matrices*, Linear Algebra Appl., 202 (1994), pp. 163–192.
[14] I. Gohberg and V. Olshevsky, *Fast algorithms with preprocessing for matrix-vector multiplication problems*, J. Complexity, 10 (1994), pp. 411–427.
[15] G. Hanrot, M. Quercia, and P. Zimmermann, *The middle product algorithm. I*, Appl. Algebra Engrg. Comm. Comput., 14 (2004), pp. 415–438.
[16] C.-P. Jeannerod and C. Mouilleron, *Computing specified generators of structured matrix inverses*, Proceedings of ISSAC'10, ACM, 2010, pp. 281–288.
[17] T. Kailath, S. Y. Kung, and M. Morf, *Displacement ranks of a matrix*, Bull. Amer. Math. Soc. (N.S.), 1 (1979), pp. 769–773.
[18] T. Kailath, S. Y. Kung, and M. Morf, *Displacement ranks of matrices and linear equations*, J. Math. Anal. Appl., 68 (1979), pp. 395–407.
[19] E. Kaltofen, *Asymptotically fast solution of Toeplitz-like singular linear systems*, Proceedings of ISSAC'94, ACM, 1994, pp. 297–304.
[20] E. Kaltofen, *Analysis of Coppersmith's block Wiedemann algorithm for the parallel solution of sparse linear systems*, Math. Comp., 64 (1995), pp. 777–806.
[21] E. Kaltofen and D. Saunders, *On Wiedemann's method of solving sparse linear systems*, in AAECC-9, vol. 539 of Lecture Notes in Computer Science, Springer, 1991, pp. 29–38.
[22] P. Lancaster and M. Tismenetsky, *The Theory of Matrices, Second Edition*, Computer Science and Applied Mathematics, Academic Press, 1985.
[23] F. Le Gall, *Powers of tensors and fast matrix multiplication*, in Proceedings of ISSAC'14, ACM, 2014, pp. 296–303.
[24] M. Morf, *Fast Algorithms for Multivariable Systems*, PhD thesis, Dept. of Electrical Engineering, Stanford University, Stanford, 1974.
[25] M. Morf, *Doubling algorithms for Toeplitz and related equations*, in Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, 1980, pp. 954–959.
[26] V. Olshevsky and V. Pan, *A unified superfast algorithm for boundary rational tangential interpolation problems and for inversion and factorization of dense structured matrices*, in Proc. 39th IEEE FOCS, 1998, pp. 192–201.
[27] V. Olshevsky and A. Shokrollahi, *A unified superfast algorithm for confluent tangential interpolation problem and for structured matrices*, in Advanced Signal Processing Algorithms, Architectures, and Implementations, ASPAAI'IX, SPIE, 1999, pp. 312–323.
[28] V. Olshevsky and A. Shokrollahi, *Matrix-vector product for confluent Cauchy-like matrices with application to confluent rational interpolation*, in STOC'00, ACM, 2000, pp. 573–581.
[29] V. Y. Pan, *Trilinear aggregating with implicit canceling for a new acceleration of matrix multiplication*, Comp. & Maths. with Appls., 8 (1982), pp. 23–34.
[30] V. Y. Pan, *On computations with dense structured matrices*, Math. Comp., 55 (1990), pp. 179–190.
[31] V. Y. Pan, *A unified superfast divide-and-conquer algorithm for structured matrices.* MSRI Preprint 1999-033, Mathematical Sciences Research Institute, Berkeley, CA, April 1999.
[32] V. Y. Pan, *Nearly optimal computations with structured matrices*, in SODA'00, ACM, 2000, pp. 953–962.
[33] V. Y. Pan, *Structured Matrices and Polynomials*, Birkhäuser Boston Inc., 2001.
[34] V. Y. Pan, *Transformations of matrix structures work again*, Linear Algebra Appl., 465 (2015), pp. 107–138.
[35] V. Y. Pan and X. Wang, *Inversion of displacement operators*, SIAM J. Matrix Anal. Appl., 24 (2003), pp. 660–677.
[36] V. Y. Pan and A. Zheng, *Superfast algorithms for Cauchy-like matrix computations and extensions.*, Linear Algebra Appl., 310 (2000), pp. 83–108.
[37] A. Schönhage, *Schnelle Multiplikation von Polynomen über Körpern der Charakteristik 2*, Acta Inform., 7 (1977), pp. 395–398.
[38] A. Schönhage and V. Strassen, *Schnelle Multiplikation großer Zahlen*, Computing, 7 (1971), pp. 281–292.
[39] I. S. Sergeev, *Fast algorithms for elementary operations with complex power series*, Diskret. Mat., 22 (2010), pp. 17–49.
[40] V. Strassen, *Gaussian elimination is not optimal*, Numer. Math., 13 (1969), pp. 354–356.

Appendix A. Proof that $\mathcal{L}(\mathsf{A})$ and $\mathcal{L}'(\mathsf{A}^{-1})$ have the same rank for $\mathcal{L}$ and $\mathcal{L}'$ as in (1). Consider first the Sylvester case, where $\mathcal{L}(\mathsf{A}) = \mathsf{MA} - \mathsf{AN}$ and $\mathcal{L}'(\mathsf{A}^{-1}) = \mathsf{NA}^{-1} - \mathsf{A}^{-1}\mathsf{M}$. If $\mathcal{L}(\mathsf{A}) = \mathsf{GH}^t$, then pre- and post-multiplying both sides of this equality by A^{-1} gives $\mathsf{A}^{-1}\mathsf{M} - \mathsf{NA}^{-1} = \mathsf{A}^{-1}\mathsf{GH}^t\mathsf{A}^{-1}$, so that $\mathcal{L}'(\mathsf{A}^{-1})$ equals $-\mathsf{A}^{-1}\mathsf{G}(\mathsf{A}^{-t}\mathsf{H})^t$ and thus has the same rank as $\mathcal{L}(\mathsf{A})$.

Consider now the Stein case, where $\mathcal{L}(\mathsf{A}) = \mathsf{A} - \mathsf{MAN}$ and $\mathcal{L}'(\mathsf{A}^{-1}) = \mathsf{A}^{-1} - \mathsf{NA}^{-1}\mathsf{M}$. Defining the matrix

$$\mathsf{B} = \begin{bmatrix} \mathsf{A} & \mathsf{M} \\ \mathsf{N} & \mathsf{A}^{-1} \end{bmatrix},$$

we have

$$\begin{bmatrix} \mathbb{I} & -\mathsf{MA} \\ & \mathbb{I} \end{bmatrix} \mathsf{B} \begin{bmatrix} \mathbb{I} & \\ -\mathsf{AN} & \mathbb{I} \end{bmatrix} = \operatorname{diag}\left(\mathsf{A} - \mathsf{MAN}, \mathsf{A}^{-1}\right)$$

and

$$\begin{bmatrix} \mathbb{I} & \\ -\mathsf{NA}^{-1} & \mathbb{I} \end{bmatrix} \mathsf{B} \begin{bmatrix} \mathbb{I} & -\mathsf{A}^{-1}\mathsf{M} \\ & \mathbb{I} \end{bmatrix} = \operatorname{diag}\left(\mathsf{A}, \mathsf{A}^{-1} - \mathsf{NA}^{-1}\mathsf{M}\right).$$

Since the four matrices applied to B are invertible, we deduce that

$$\operatorname{rank}(\mathsf{B}) = \operatorname{rank}(\mathsf{A} - \mathsf{MAN}) + \operatorname{rank}(\mathsf{A}^{-1}) = \operatorname{rank}(\mathsf{A}) + \operatorname{rank}(\mathsf{A}^{-1} - \mathsf{NA}^{-1}\mathsf{M}).$$

Using $\operatorname{rank}(\mathsf{A}) = \operatorname{rank}(\mathsf{A}^{-1})$, we see again that $\mathcal{L}(\mathsf{A})$ and $\mathcal{L}'(\mathsf{A}^{-1})$ have the same rank.

Appendix B. Proof of Lemma 18. Fix i and j. For all $\ell \geqslant 1$, [35, Theorem 4.7] gives

$$\mathsf{A}'_{i,j} - \mathbb{M}^{\ell}_{P_i}\, \mathsf{A}'_{i,j}\, (\mathbb{M}^t_{Q_j})^{\ell} = \sum_{k \leqslant \alpha} \mathbb{K}(\mathbb{M}_{P_i}, \mathsf{g}_{i,k}, \ell)\, \mathbb{K}(\mathbb{M}_{Q_j}, \mathsf{h}_{j,k}, \ell)^t$$

with $\mathbb{K}(\mathbb{M}_{P_i}, \mathsf{g}_{i,k}, \ell)$ and $\mathbb{K}(\mathbb{M}_{Q_j}, \mathsf{h}_{j,k}, \ell)$ as in the proof of Lemma 15. Writing $Q_j = q_{j,0} + \cdots + q_{j,n_j} x^{n_j}$, and multiplying the former equality on the left by $q_{j,\ell}\, \mathbb{M}^{n_j-\ell}_{P_i}$, we get, for $1 \leqslant \ell \leqslant n_j$,

$$\begin{aligned} &q_{j,\ell}\, \mathbb{M}^{n_j-\ell}_{P_i}\, \mathsf{A}'_{i,j} - q_{j,\ell}\, \mathbb{M}^{n_j}_{P_i}\, \mathsf{A}'_{i,j}\, (\mathbb{M}^t_{Q_j})^{\ell} \\ &\quad = \sum_{k \leqslant \alpha} \mathbb{M}^{n_j-\ell}_{P_i} \mathbb{K}(\mathbb{M}_{P_i}, \mathsf{g}_{i,k}, \ell)\, q_{j,\ell}\, \mathbb{K}(\mathbb{M}_{Q_j}, \mathsf{h}_{j,k}, \ell)^t \\ &\quad\quad = \sum_{k \leqslant \alpha} \mathbb{K}(\mathbb{M}_{P_i}, \mathsf{g}_{i,k}, n_j)\, \mathbb{J}_{n_j}\, q_{j,\ell}\, \mathbb{J}_{\ell,n_j}\, \mathbb{K}(\mathbb{M}_{Q_j}, \mathsf{h}_{j,k}, n_j)^t, \end{aligned}$$

since the last ℓ columns of $\mathbb{K}(\mathbb{M}_{P_i}, \mathsf{g}_{i,k}, n_j)$ are precisely $\mathbb{M}^{n_j-\ell}_{P_i}\, \mathbb{K}(\mathbb{M}_{P_i}, \mathsf{g}_{i,k}, \ell)$; note that the equality also holds for $\ell = 0$. Summing over all $\ell = 0, \ldots, n_j$, and using the fact that $Q_j(\mathbb{M}^t_{Q_j}) = 0$, we deduce

$$\widetilde{Q_j}(\mathbb{M}_{P_i})\, \mathsf{A}'_{i,j} = \sum_{k \leqslant \alpha} \mathbb{K}(\mathbb{M}_{P_i}, \mathsf{g}_{i,k}, n_j)\, \mathbb{J}_{n_j}\, \mathbb{Y}_{Q_j}\, \mathbb{K}(\mathbb{M}_{Q_j}, \mathsf{h}_{j,k}, n_j)^t.$$

The rest of the proof now follows exactly that of Lemma 15.